DEVELOPMENT OF PRECONDITIONED CONJUGATE GRADIENT METHO FOR NON-LINEAR DIFFUSION PROBLEMS IN COMPLEX GEOMETRIES

by

AJAY MOHAN SHEJUL

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

MARCH, 1995

DEVELOPMENT OF PRECONDITIONED CONJUGATE GRADIENT METHOD FOR NON-LINEAR DIFFUSION PROBLEMS IN COMPLEX GEOMETRIES

A thesis submitted
in Partial Fulfillment of the Requirements
for the Degree of

M A S T E R   O F   T E C H N O L O G Y

by

AJAY MOHAN SHEJUL

to the

Department of Mechanical Engineering

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

MARCH, 1995.

ME-1995-M-SHE-DEV

CERTIFICATE

It is certified that the work contained in the thesis entitled **DEVELOPMENT OF PRECONDITIONED CONJUGATE GRADIENT METHOD FOR NON-LINEAR DIFFUSION PROBLEMS IN COMPLEX GEOMETRIES,** by **AJAY MOHAN SHEJUL,** has been carried out under my supervision and that this has not been submitted else where for a degree.

K Muralidhar

K. MURALIDHAR
Dept. of Mechanical Engg.
I.I.T., KANPUR

March, 1995.

ACKNOWLEDGEMENTS

I express my deep sense of gratitude to my guide Dr. K. Muralidhar for the untiring guidance and unceasing encouragement without which this work wouldn't have been possible.

I would like to express my thanks to Mr. P. M. V. Subbarao and K. Mahesh who helped me innumerable number of times throughout my thesis work. I would also like to thank all my friends and specially my family for the tremendous encouragement they gave me at every point of time

Ajay Mohan Shejul

SYNOPSIS

Matrix inversion is one of the most computationally intensive steps during the numerical solution of partial differential equations. Direct matrix inverters involve a greater number of arithmetic operations and hence more round off errors. However they are generally stable. In contrast to this, iterative methods can solve a system of equations with less effort, though they are only conditionally stable. The memory requirement of iterative method is less than the direct method. To combine the benefits of both approaches, hybrid methods are presently being developed. Preconditioned Conjugate Gradient Method (PCGM) is one such technique that has been considered in the present study.

Specific advantages of the PCG technique includes rapid convergence of the iterations, reduced memory requirement and the possibility of parallelization on multi-processor computers.

Two different preconditioners have been developed in the present work. They have been compared with the performance of a sparse matrix solver based on Gaussian elimination. The PCG algorithm has been applied to the following problems.

1. Steady and unsteady heat conduction in a complex geometry with temperature dependent conductivity.
2. Flow of a viscoplastic power law fluid in a rock fracture including contact areas.
3. Displacement of oil in a rock fracture by high pressure injected water.

Studies clearly show the superiority of PCG over conventional techniques. Between the two preconditioners developed in the study, the one based on LU decomposition of a simpler problem is found to yield excellent results in terms of monotone convergence, CPU time and computer memory.

CONTENTS

LIST OF FIGURES

LIST OF TABLES.

NOMENCLATURE

L	Lower triangular matrix.
U	Upper triangular matrix.
D	Diagonal matrix.
T	Temperature, Dimensionless form.
x y	Cartesian Coordinates, m
L	Length of the domain, m
K	Thermal Conductivity, W/m K
t	Time, s
Ni	Shape function.
Ω	Domain.
$\partial\Omega$	Boundary of the domain.
NE	Number of elements.
NOB	Number of obstacles.
u_o, v_o	Velocity of oil in x and y directions, m/s
u_w, v_w	Velocity of water in x and y directions, m/s
ρ_o, ρ_w	Density of oil and water respectively, kg/m^3
ψ	Body force per unit volume, N/m^3
p_o, p_w	Oil and water pressure respectively, N/m^2
ξ_o, ξ_w	Compressibility of oil and water respectively, Pa^{-1}
β_o, β_w	Volumetric thermal expansion of oil and water, K^{-1}
n	Power law index.
K_o	Consistency index in Herschel Bulkely model, Pa s^n
K_w	Viscocity of water, Pa.s
τ	Shear stress, N/m^2
τ_o	Yield shear stress, N/m^2
S_o, S_w	Saturation of oil and water.
p_c	Capillary pressure, N/m^2
K_{ro}, K_{rw}	Relative permeability of oil and water.
ppv	Percentage pore volume.
2h	Fracture height, m
η_x, η_y	Direction cosine in x and y direction.

CHAPTER 1
INTRODUCTION

1.1 Preliminary Remarks

In the analysis of fluid systems one generally comes across complicated nonlinear differential equations. The origin of nonlinearity of these equations arises from intrinsic fluid properties, choice of co-ordinate systems, complicated geometries, heterogeneous medium, complicated boundaries and complicated boundary conditions. The exact solution of such equations may be difficult to determine or may not exist in the classical sense. Exact solutions, when possible for a simpler hypothetical problems, are not useful for practical problems. Solutions to complicated problems are possible either by doing experiments or by using numerical techniques. Performing experiments requires money, time and labour. On the other hand numerical analysis is much cheaper. Computation along with experiments is the best tool of solving any problem.

Large scale applications in engineering that require numerical analysis include Weather forecasting, Oil reservoir simulation, Finite element analysis of composite structures and Aerodynamics, Commonly used numerical techniques are Finite differences, Finite element method and Spectral method. These methods invariably reduce the governing differential equation to a set of algebraic equations. These equations can be represented in the form of a matrix. The main task of finding the solution involves the inversion of the matrix.

Large scale applications in three dimensions generate large matrices. Inversion of such matrices requires substantial memory and a large number of arithmetic operations need to be performed. In order to minimize computational time as well as round off error the matrix inversion should be done with fewer number of arithmetic operations. Finding optimal matrix inverters is a topic of current research.

Matrix inverters are classified into two major groups, (a) Direct Methods (b) Iterative Methods. Direct methods are unconditionally stable but they are computationally intensive. These methods are especially unsuitable for non linear equations. In such problems every non-linear iteration requires a new matrix to be constructed. It has to be inverted repeatedly and the computing cost can become quite high. The other disadvantage of this method is that the full matrix is required to be formed. These methods are particularly suitable when the matrix is banded and the bandwidth is small. However for smaller bandwidth, specialized algorithms have been developed that require fewer arithmetic operations and less computational time. Various direct methods available are Gauss elimination technique, Thomas tridiagonal algorithm and Block diagonal methods.

Though Gaussian elimination does not require diagonal dominance, it cannot handle zeros on the diagonal of matrix. This is not a serious limitation because the diagonal entries of the matrix can be made non-zero by suitably swapping rows and columns. Of greater concern however is the build-up of round-off errors. Since the diagonal entries of the matrix to be inverted are repeatedly used, very small values will be affected by round-off errors introduced by a computer. This problem is compounded when large systems of equations are encountered. A partial remedy for this problem is the use of a pivoting strategy. The pivoting operation comprises of swapping rows and columns till the diagonal terms are a maximum compared to all other terms present in their respective row and column. When similar matrices are to be repeatedly inverted,the sequence of swapping operations can be stored once-and-for-all. This will greatly reduce the computational cost of pivoting. Swapping columns of a matrix will generate new right hand side vectors and is hence an expensive task. In many problems swapping rows alone can improve the situation and control round-off. This is called as partial pivoting. When pivoting does not make diagonal entries of the matrix large enough, the question of round-off remains critical and the only solution available is to generate matrices which are themselves not very large. This can be accomplished in practical applications by domain decomposition and in multidimensional problems by reducing them to a sequence of one dimensional problems.

In case of the iterative methods an initial guess for the solution vector is required to be given at the start of the algorithm. Iterative methods are only conditionally stable. The advantage of this method is that full matrix need not be formed and explicitly stored. The rate of convergence of these methods depends on the initial guess and also on the size of the matrix. If the size of matrix is large, this method performs poorly. A major constraint in most iterative techniques is the need for diagonal dominance. They have however been extensively used for problems of intermediate complexity. The traditional iterative methods are those of Jakobi and Gauss-Seidel.

The advantages of Direct methods are,

(a) They are unconditionally stable.

(b) They do not require diagonal dominance.

Advantages of iterative methods are

(a) Full matrix need not be formed explicitly.

(b) They converge rapidly when provided with a good initial guess.

In order to have the advantages of both iterative and direct inversion techniques, hybrid inversion techniques are being developed. Some of these methods are Incomplete Cholesky decomposition, Alternating direction implicit method. Preconditioned conjugate gradient method and Matrix partitioning.

These methods have the advantage that the matrix is not formed completely and arithmetic operations are just vector multiplications. With the invention of vector and parallel computer these methods have gained importance because they are particularly suitable for such machines. Strong convergence results are available for many of these methods and it is now possible to invert large matrices with fewer number of operations.

The Conjugate Gradient algorithm, developed originally for symmetric positive definite matrix is an example of such a method. With preconditioning it can now be used for unsymmetric and possibly ill-conditioned matrices. Unfortunately this iterative method for coupled Navier stokes equation has been handicapped by the lack of good preconditioner techniques that can adequately treat the dominating effect of the incompressibility constraint. Preconditioners tend to be problem dependent. The present study is restricted to developing and testing Preconditioned Conjugate Gradient method for a nonlinear transient diffusion-dominated problem.

1.2 Literature Survey

A review of literature shows that the conjugate gradient algorithm for solving linear systems of equations and eigenvalue problems represent very important computational innovations of the early 1950s. These methods came into wide use only in the mid seventies. Shortly thereafter,vector computers and massive computer memories made it possible to use methods to solve problems which could not be solved in any other way. Since that time the algorithms have been further refined and have become a basic tool for solving a wide variety of problems on various computer architecture. The Conjugate Gradient algorithm has also been extended to solve non-linear systems of equations and optimization problems. This has had tremendous impact on the computation of unconstrained and constrained optimization problems.

Golub and O'leary [1989] have given some of the history of the Conjugate Gradient Algorithm during the period from their original development to their widespread application in the mid 1970s. The original development of this algorithm was carried out by a core of researchers including *C. Lanczos and M. Hestenes* at the Institute for Numerical Analysis in the National Applied Mathematics Laboratories at the United States National Bureau of standards (NBS) in Los Angeles and *E. Stiefel* of the Eidg. Technische Hockshule Zurich. The first communication among the NBS researchers and *Stiefel* concerning this algorithm seems to have been at a symposium on simultaneous Linear equations and the Determination of eigenvalues, held at INA in August 1951, as discussed in *Stiefel* (1952).

The first presentation of conjugate direction algorithm is by *Fox, Huskey and Wilkinson* (1948) who considered them as a direct methods, and by *Forsythe, Hestenes, and Rosser (*1951)*,Hestenes and Stiefel* (1952) *and Rosser* (1953), who discuss the NBS research. *Hestenes, Lanczos and Stiefel* clearly considered the algorithm to be a full **N** step direct method for certain problems, but also suggested its use as an iterative algorithm requiring fewer than **N** steps for well conditioned problems and possibly more than **N** steps for ill-conditioned ones. Computational results for the Conjugate Gradient Algorithm were presented in *Stiefel* (1953), *Fischbach* (1956) *and Stiefel* (1958).

In 1960s, the Conjugate Gradient Algorithm began to acquire a mixed reputation. *Frank* (1960) tested the algorithm on a matrix with eigenvalues related to the Chebyshev polynomials, the hardest case for Conjugate Gradients and reported slow convergence. *Livesley* (1960) applied it to structural analysis and was unsuccessful. *Campbell* (1965) studied ocean circulation and *Wilson* (1966) solved optimal control problems with the aid of

Conjugate Gradients.

Ideas leading to successful preconditioned conjugate gradient algorithms were subsequently developed. *Dufour* (1964) applied conjugate gradients to problems in geodesy and discussed several important ideas, including extensions to least squares problems and preconditioning by equality constraints. *Varga* (1960) suggested a sparse partial factorization of a matrix as a splitting operator for Chebyshev acceleration.

Although Conjugate Gradient Algorithms were widely used in 1960s, the numerical analysis community was not satisfied with the understanding of algorithm and its speed. Preconditioning techniques were not widely known and it was in the early 1970s that key developments were made in making preconditioned algorithms practical. The paper of *Reid* (1971) drew the attention of many researcher to the potential of the algorithm as an iterative method for sparse linear systems. It was a catalyst for much of the subsequent work in Conjugate Gradients.

Preconditioning techniques, although discussed in the 1960s have now become widely used. *Axelsson* (1972) suggested Preconditioning Conjugate Gradients by a scaled SSOR operator. Other preconditioning strategies were discussed by *Evans* (1973). *Bartels and Daniel* (1974) *Chandra, Eisenstat and Schultz* (1975), *Concus, Golub and O'Leary* (1976), *Douglas and Duport* (1976) *Meijerink and Van der Vorst* (1977) and *Kershaw* (1977), *Concus, P. and Meurant, G* (1986).

1.3 Preconditioning

The word *preconditioning* was used by *Turing* (1948) and since then, it has become a standard terminology for problem transformation in order to make the solution easier. *Evans* (1973) made use of preconditioning for improving the convergence of an iterative Conjugate Gradient method. The matrix which is neither symmetric nor positive definite can be inverted by Preconditioned Conjugate Gradient Method under certain restrictive conditions. Even if this is possible, the Preconditioned Conjugate Gradient Algorithm will be hampered because of increased round-off errors as well as increased iterations for convergence. The rate of convergence depends upon the spectral condition number of the matrix. If the spectral condition number is closed to unity convergence is rapid. Hence the goal of preconditioning can be interpreted as decreasing the spectral condition number towards unity.

1.4 Scope of present work

The present work is in two parts. In the first part, the Preconditioned Conjugate Gradient method is studied and two different preconditioners have been developed. The two preconditioners are compared with the Gauss elimination method in terms of CPU time and memory requirement. For this purpose two test problems have been considered, namely, (a) steady heat conduction and (b) Nonlinear unsteady heat conduction, both in complex geometries. Among the two **PCG** methods and Gaussian elimination, one method is selected as the best and applied to oil recovery from a fracture.

The second part of the thesis concerns simulation of oil recovery by water injection. The mathematical modeling of the problem results in two highly non-linear coupled partial differential equation, each for oil pressure and water pressure. In all problems studied, matrices to be inverted by **PCG** are generated by the finite element method.

CHAPTER 2

PRECONDITIONED CONJUGATE GRADIENT METHOD

2.1 Introduction

The Conjugate Gradient method combines features of both iterative and direct solvers. It requires an initial guess to start the iterations. In the absence of round off errors it converges in a maximum of **N** iterations, where **N** is the size of the matrix. The original **CGM** was used only for solving symmetric positive definite matrices. Through preconditioners it can however be used for unsymmetric matrices as well. The Conjugate Gradient method converges faster than the Gauss-Seidel method for a given initial guess but needs more arithmetic operations per iteration. This method minimizes an error functional along mutually conjugate directions in each iteration. Hence for a functional defined in N dimensions, it converges theoretically in **N** iterations. For a matrix with a small spectral condition number, convergence is attained in just a few iterations, much less than N. This method is particularly suitable for sparse matrices and for Finite element analyses. While computing the solution of a large problem, large matrices can also be avoided in the sense that a global matrix need not be constructed. This reduces computer memory requirement.

2.2 The Conjugate Gradient Method

The original Conjugate Gradient method without preconditioning is given below.

The matrix equation is $Ax = b$

1. Assume $x = x^o$ (initial guess)

2. Compute residue

$$r^k = b - Ax^k$$

3. Set the direction vector

$$p^k = r^k$$

4. Compute steepest descent parameter, α

$$\alpha = \frac{(r^k, r^k)}{(p^k, Ap^k)}$$

5. Update the values of x and r

$$x^{k+1} = x^k + \alpha .p^k$$

$$r^{k+1} = r^k - \alpha .Ap^k$$

6. Compute β

$$\beta = \frac{(r^{k+1}, r^{k+1})}{(r^k, r^k)}$$

7. Update direction vector

$$p^{k+1} = r^{k+1} + \beta p^k$$

8. Check for convergence in x.

9. Repeat steps 4-8 until convergence.

2.3 Need for Preconditioning

If the matrix to be inverted by Conjugate Gradient is not symmetric and positive definite, preconditioning is required. Even for matrices that satisfy this condition, preconditioning can be used to improve the rate of convergence. The rate of convergence depends upon the spectral condition number of the matrix. The spectral condition number is defined as the ratio of maximum eigenvalue to the minimum eigenvalue of the matrix. If the spectral condition number is close to unity, the rate of convergence of the Conjugate Gradient iterations is fast. This means that eigenvalues must cluster around a certain number. This is the stated goal of preconditioning.

Preconditioning involves the choice of a suitable positive definite matrix **M** so that $Ax = b$ may be replaced by $\mathbf{M}^{-1}\mathbf{Ax} = M^{-1}b$, where $M^{-1}A$ should have a large number of its eigenvalues nearly equal or closely grouped. Three preconditioning strategies are described below.

2.3.1 Preconditioning I (Symmetric Matrices)

The equation **Ax=b** is written as $\mathbf{(LL^T)^{-1} Ax = (LL^T)^{-1}b}$where **L** is the lower triangular matrix, which may be computed efficiently by Cholesky decomposition. The Cholesky method is not suitable for an asymmetric matrix **A**, since it cannot be factored as $\mathbf{LL^T}$. If nonzero entries of **L** are computed at those position which correspond to nonzero elements in the lower triangle of **A** then decomposition is incomplete. If we solve the matrix equation using the approximate **L** matrix the Conjugate Gradient method is called Incomplete Cholesky Conjugate Gradient Algorithm (ICCG). The complete algorithm of ICCG is given below:

1. Set $k = 0$ and $x = x^o$ (initial guess)
2. Compute $r^k = b - Ax^k$
3. Compute direction vector
$$p^k = (LL^T)^{-1}r^k$$
4. Compute steepest descent parameter, $\propto$
$$\alpha = \frac{(r^k, (LL^T)^{-1}r^k)}{(p^k, Ap^k)}$$
5. Update the values of x and k

$$x^{k+1} = x^k + \alpha\, p^k$$

$$r^{k+1} = r^k - \alpha . Ap^k$$

6. Compute β

$$\beta = \frac{(r^{k+1},\ (LL^T)^{-1}\ r^{k+1})}{(r^k,\ (LL^T)^{-1} r^k)}$$

7. Update direction vector

$$p^{k+1} = (LL^T)^{-1}\ r^{k+1} + \beta . p^k$$

8. Set k = k+1, check for convergence in x.

9. Repeat steps 4-8 until convergence.

2.3.2 Preconditioning II (Asymmetric Matrices)

For asymmetric non singular matrices the Cholesky decomposition is replaced by the LU decomposition where we take unit diagonal elements in the upper triangular matrix **U**. If **L** and **U** have the same structure as the lower and upper parts of **A** then the resulting **PCG** method is called Incomplete LU decomposition Conjugate Gradient Algorithm (ILUCG). The complete algorithm of ILUCG is as follows,

1. Assume

$$x = x^o \text{ (initial guess)}$$

2. Compute residue

$$r^k = b - Ax^o$$

3. Compute direction vector

$$p^k = (U^TU)^{-1} A^T\ (LL^T)^{-1}\ r^k$$

4. Compute steepest descent parameter, α

$$\alpha = \frac{(r^k,\ (LL^T)^{-1} r^k)}{(p^k,\ (U^TU)\ p^k)}$$

5. Update the values of x and r

$$x^{k+1} = x^k + \alpha . p^k$$

$$r^{k+1} = r^k - \alpha . Ap^k$$

6. Compute β

$$\beta = \frac{(r^{k+1},\ (LL^T)^{-1}\ r^{k+1})}{(r^k,\ (LL^T)^{-1} r^k)}$$

7. Update the direction vector

$$p^{k+1} = ((U^TU)^{-1}A^T(LL^T)^{-1})\ r^{k+1} + \beta . p^k$$

8. Set k = k+1; check for convergence in x.

9. Repeat steps 4-8 till convergence.

2.3.3 Diagonal Scaling

A simple preconditioned strategy is based on scaling the matrix A, so that the diagonal entries are strengthened with respect to other element within the matrix. This algorithm is given below. The equation **A x** = b is written as

$$D^{1/2}\ AD^{-1/2}\ D^{1/2}x = D^{-1/2}b$$

i.e.

$$\bar{\mathbf{A}}\bar{\mathbf{x}} = \bar{\mathbf{b}}$$

where

$$\bar{\mathbf{A}} = D^{-1/2}AD^{-1/2}$$

$$\bar{\mathbf{x}} = D^{1/2}x$$

$$\bar{\mathbf{b}} = D^{-1/2}b$$

where D is the matrix containing only the diagonal entries of A. This method has not been studied in this thesis.

2.3.4 Efficient Calculation of Matrix Products and Inverses

In the present work ILUCG method is used for matrix inversion. In this method the computation of quantities such as g

$=(LL^T)^{-1}r$ will require more computation time $(O(N))^3$if the conventional Gauss elimination technique is used. An efficient method that requires only $O(N^2)$ operations is described below.

Writing $\quad g = (LL^T)^{-1}r \quad$ (2.1)

we get $\quad (LL^T)g = r \quad$ (2.2)

let $\quad (L^Tg) = f \quad$ (2.3)

then $\quad Lf = r \quad$ (2.4)

To compute vector g, we solve (2.4) for f and use f to calculate g using (2.3). This method requires just forward and backward substitutions, since L is in lower triangular form.

2.4 Description of Two Preconditioners Developed in the Study

The two preconditioning methods developed in the present work are labelled as M2 and M3. The efficiency of these two methods is compared with method M1 which is based on Gaussian elimination (*Duff*, 1980).

Method M2 uses the complete LU decomposition of the matrix derived from a less complex problem. These sub-matrices L & U are used as preconditioners for more complex problems. In the present work for steady and unsteady heat conduction problems L and U are calculated for a problem in which obstacles or inhomogeneous regions are absent. These are then used for problems having complex regions. Thus we see that L and U are computed just once and subsequently used for problems of greater complexity. We require that complex problem generates a matrix of same size as L and U. In the present work we have studied the convergence of this method by plotting the percentage residue $\left[|r|/|x|\right]^{1/2}$ x

100as a function of number of iterations.

The above method of supplying L and U does not work efficiently for nonlinear problems. It takes more iteration for convergence as the complexity of problem increases. For nonlinear problems, method M2 has been modified in the sense that L and U are supplied in a different manner. In these problems the complete L and U are computed at the first iteration of nonlinearity and these are then stored and are used for all further iterations. This way of supplying L and U is found to be efficient and convergence of residue with iterations is seen to be monotonic.

In method M3 the two matrices L and U are calculated only on the nonzero diagonals of matrix A. While using finite element method, if the element shape and number of nodes per element is fixed, one can easily determine the number of non-zero diagonals in the assembled matrix. In the present work a six-noded triangular element is used. Thus each node will be affected by adjacent 18 nodes. This is shown in Figures 2.1 and 2.2. There will be a total of 19 non-zero diagonals for any size of the grid. The bandwidth of the matrix is also calculated once the global node numbers are assigned. Thus in method M3 we see that L and U are computed only on few locations of the matrix. This is similar to ICCG, where L and U are calculated at non-zero entries of A. This reduces the computation time for calculating L and U as compared to method M2 especially as the size of the matrix increases.

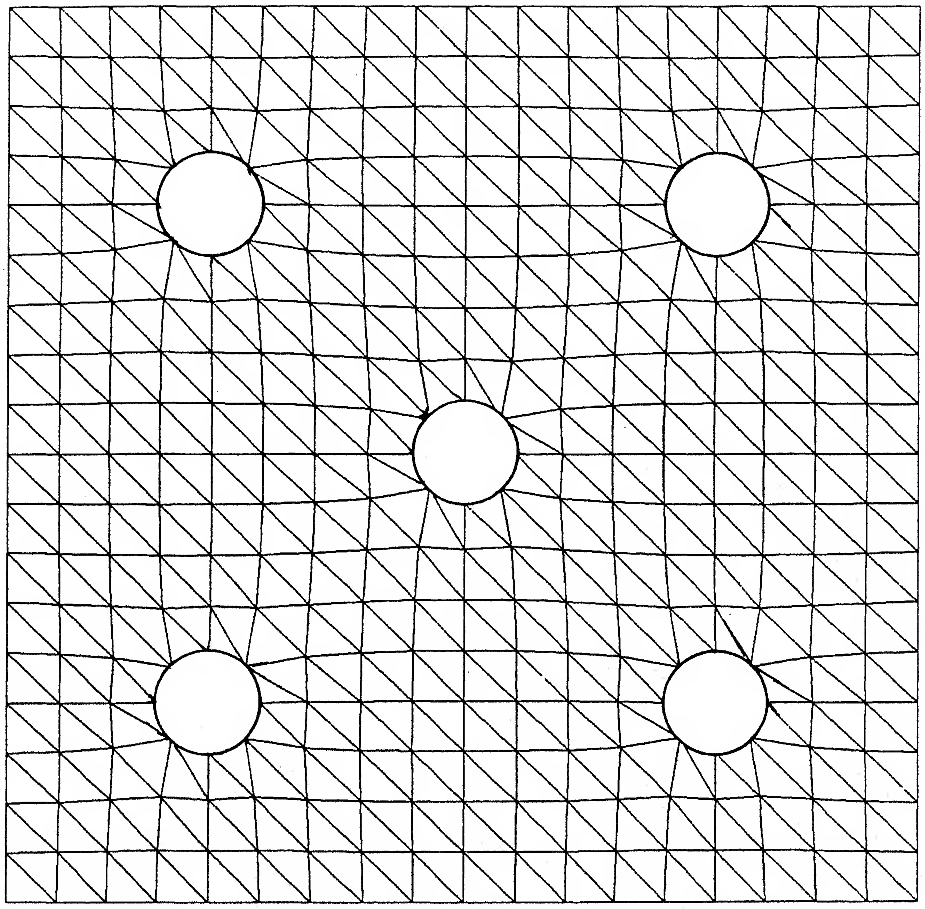


Figure 2.1 : Grid for Five Obstacles.

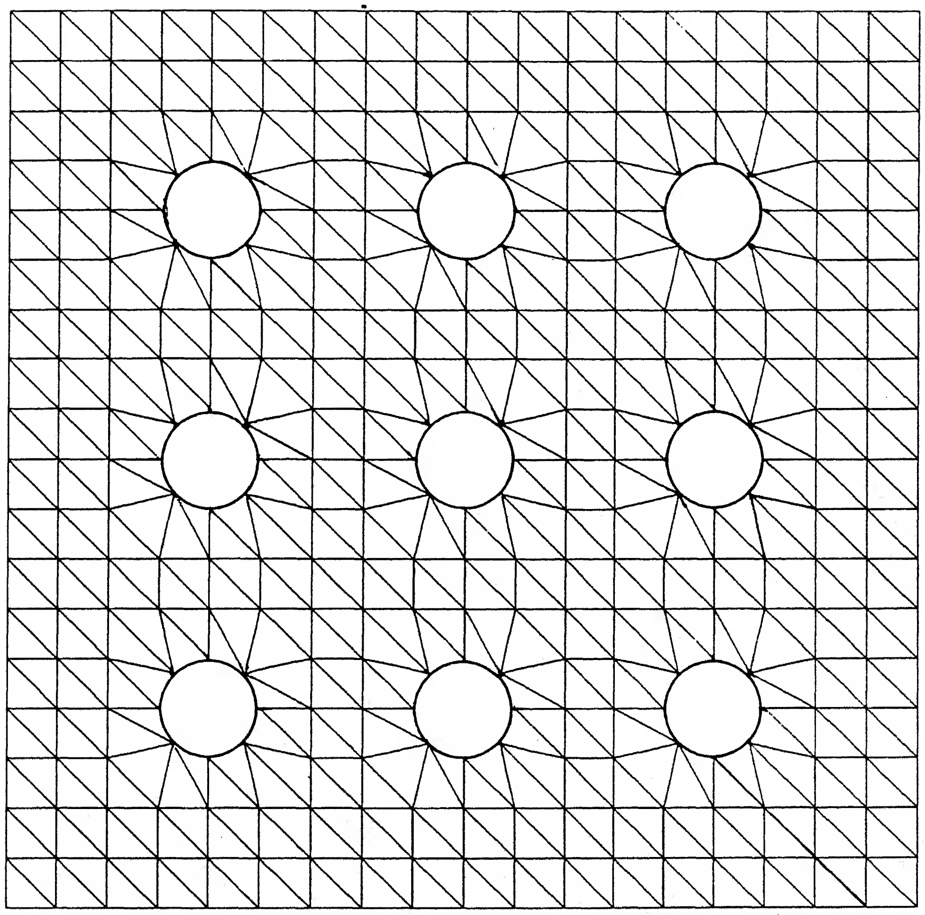


Figure 2.2 : Grid for Nine Obstacles.

Method M3 for calculating L and U does not serve as a good preconditioner. It takes more PCG iterations for inversion of the matrix. Moreover convergence is not monotonic. The number of iterations increases as the size of matrix increases. This is due to increased sparsity of L and U as the matrix size increases.

In implementation of both methods M2 and M3, all the arrays including those of matrices are one dimensional. The advantage of storing in one dimensional arrays is that less computational time is required while accessing any element within. The complete matrix can be described by three arrays. The first array stores the elements of matrix row wise only within the bandwidth. No element outside the bandwidth is stored. The second array stores the column number of each element. The third array is a pointer array. The difference between two adjacent elements gives number of elements in a particular row. This method of storing also reduces memory requirement.

The convergence criteria used in methods M2 and M3 is as follows:

$$100 \times \sqrt{\frac{ZZ}{XX}} \leq 0.01$$

where $ZZ = \Sigma r_i^2$ is the summation of residue vector

$XX = \Sigma x_i^2$ is the summation of solution vector.

The two methods M2 and M3 are compared with respect to CPU time and memory requirement in the Chapter 3.

2.5 Parallellization.

The CGM has inherent advantages during parallelization. The computational requirements of Conjugate Gradient Algorithm per iteration are (1) The matrix vector multiplication, Ap^k,

(2) The inner products (r^k, r^k) and (p^k, Ap^k),

(3) The linked triad operations involving u^{k+1}, r^{k+1} and p^{k+1},

(4) Two scalar divides for computing α and β and

(5) One scalar comparison for testing convergence

The Conjugate Gradient Algorithm can be parallelized in the following way. The variables x,r,p and Ap associated with grid points are distributed among the processors.The inner products are calculated by the processors and the results are collected by a fan in operation. Calculation of α and β are done globally and then passed to other processors. Each processor calculates the sum of the squares of residue vector and sends it to the manager processor which adds up all such products. Similarly the inner product (p^k, Ap^k) is also computed in the manager processor.The value of α is calculated and transmitted to all processors. The worker processor after receiving this value proceeds to update the values of vectors x, r and p. The residue is calculated at each processor and the manager sums them up to check for convergence.

Details of parallelization of the Conjugate Gradient Algorithm can be found in *Hackbusch* (1992). Parallelization of PCG method can be found in *Amodio and Brugnano* (1990) and *Brugnano and Trigiante* (1991).

CHAPTER 3

APPLICATION OF PCG TO STEADY HEAT CONDUCTION PROBLEM

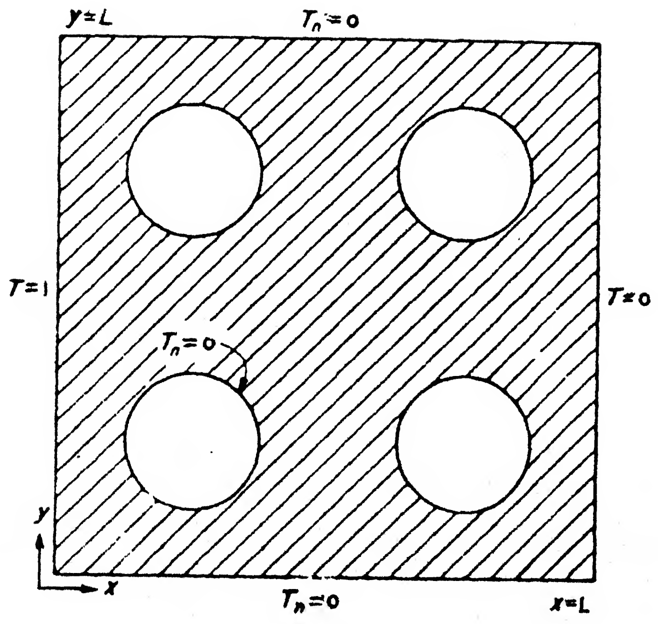
3.1 Introduction

The present chapter concerns the application of methods M1, M2, M3 described in Chapter 2 to the Finite element solution of steady heat conduction in complex geometries.

3.2 Geometric Model and Potential Use

The geometry and coordinate systems are as shown in Figure 3.1. A square region of size 6x6 dimensionless units is considered. Circular inhomogeneities are distributed in the region. The conductivity of inhomogeneities are assumed to be very less compared to the homogeneous region. In a Cartesian coordinate system x = 0 and x = L are the inflow and outflow planes of heat flux respectively. On these planes constant temperatures are specified. The upper and lower edges of domain i.e. y = 0 and y = L are insulated. On all the surfaces of the obstacles $\frac{\partial T}{\partial n} = 0$, where n is the unit outward normal. This result is due to the fact that the conductivity of inhomogeneities is very less as compared to the conductivity of the matrix. The insulated side wall boundary conditions at y = 0 and y = L can be interpreted as symmetry conditions in a large composite which has a repeating pattern of fibre distribution normal to the mean temperature gradient.

Composite materials have wide application in engineering due to their high strength to weight ratio. Various applications of



igure 3.1 : Physical Domain, Coordinate System and Boundary Conditions.

composites include Aircraft wings, Satellites and cylinder blocks of I.C. engines. The composite structure is required to withstand a large thermal loading as well. In some instances, the components which make up the composite have widely differing strengths as well as thermal conductivities. Hence while the strength is improved, the equivalent conductivity of the composite deteriorates to lower values. For a given heat flux, this can mean higher temperatures in the structures and consequently lowering of strength itself. Hence it is important to determine the effective conductivity of the composite.

3.3 Governing Differential Equations and Boundary Conditions

The equation which governs steady heat conduction in two dimensions is of the form,

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \qquad (3.1)$$

Boundary conditions employed in the present work are,

$$\begin{aligned} x &= 0 &&: \quad T = 1 \\ x &= L &&: \quad T = 0 \qquad (3.2) \\ y &= 0,\ L &&: \quad \frac{\partial T}{\partial y} = 0 \end{aligned}$$

Equations 3.1 and 3.2 have been solved using a Galerkin finite element method. The element level matrix is obtained by weighted integration of the governing equation over any element(Appendix-A). This equation is

$$\int_{\Omega^e} \frac{\partial N_i}{\partial x} \cdot \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \cdot \frac{\partial N_j}{\partial y} = \int_{\partial\Omega^e} N_i \frac{\partial T}{\partial n} \cdot ds \qquad (3.3)$$

For the boundary conditions given by Equation 3.2 the right hand side of (3.3) will become

$$\int_{\partial\Omega^e} N_i \frac{\partial T}{\partial n} = 0$$

The finite element solution proceeds as follows. The whole domain is divided into smaller triangular elements each having six nodes. The elements are chosen as isoparametric. This allows exact representation of circular boundaries of the contact areas. The grid is generated within as well as outside the contact area but with the shape preserved. Eight elements lie within one obstacle and each obstacle surface is discretized by at least 8 nodes. The elements which lie within the obstacle are assumed to have a conductivity 1.5E-04 while the matrix has unit conductivity.

3.4 Comparison of Results using Methods M1,M2 and M3

In this section a comparison of three methods M1, M2, M3 to solve the matrix equation arising in **FEM** is carried out with respect to the following factors:

a. Accuracy

b. CPU time

c. Memory requirement

d. Method M2 and M3 are compared with respect to their convergence.

3.4.1 Accuracy

Table 3.1 shows the results obtained by the three methods. Table 3.1 gives the values of temperature at various x locations while y is kept constant at y = 3, the midplane. From Table 3.1

Table 3.1 Temperature Profiles Computed by Methods $M1$, $M2$ and $M3$

NE = 288	*NOB = 1*			*NE = 288*	*NOB = 5*		
x	$M1$	$M2$	$M3$	x	$M1$	$M2$	$M3$
.0000	1.0000	1.0000	1.0000	.0000	1.0000	1.0000	1.0000
.2560	.9599	.9599	.9599	.2650	.9616	.9616	.9615
.5120	.9199	.9199	.9199	.5300	.9225	.9225	.9224
.7680	.8801	.8801	.8800	.7880	.8833	.8833	.8832
1.0240	.8405	.8405	.8405	1.0460	.8425	.8426	.8425
1.2800	.8015	.8015	.8015	1.2860	.8037	.8037	.8036
1.5370	.7633	.7633	.7632	1.5260	.7644	.7644	.7643
1.7900	.7270	.7271	.7269	1.7690	.7261	.7261	.7260
2.0430	.6930	.6930	.6928	2.0110	.6906	.6906	.6906
2.3170	.6637	.6636	.6635	2.2960	.6576	.6576	.6576
2.5900	.6486	.6485	.6484	2.5810	.6402	.6402	.6402
2.8060	.5691	.5727	.5690	2.8010	.5655	.5681	.5656
3.0220	.4922	.5022	.4920	3.0200	.4935	.5023	.4936
3.2180	.4226	.4291	.4225	3.2210	.4276	.4336	.4276
3.4140	.3510	.3511	.3509	3.4230	.3595	.3595	.3595
3.6700	.3375	.3375	.3373	3.6930	.3434	.3434	.3434
3.9260	.3108	.3108	.3107	3.9630	.3129	.3129	.3129
4.1870	.2760	.2760	.2759	4.2130	.2767	.2767	.2768
4.4490	.2387	.2387	.2386	4.4630	.2373	.2373	.2373
4.7090	.2001	.2001	.2000	4.7060	.1976	.1976	.1976
4.9690	.1605	.1606	.1605	4.9490	.1583	.1583	.1582
5.2270	.1207	.1207	.1206	5.2080	.1173	.1173	.1173
5.4850	.0806	.0806	.0805	5.4680	.0780	.0779	.0779
5.7430	.0403	.0403	.0403	5.7340	.0386	.0386	.0386
6.0000	.0000	.0000	.0000	6.0000	.0000	.0000	.0000

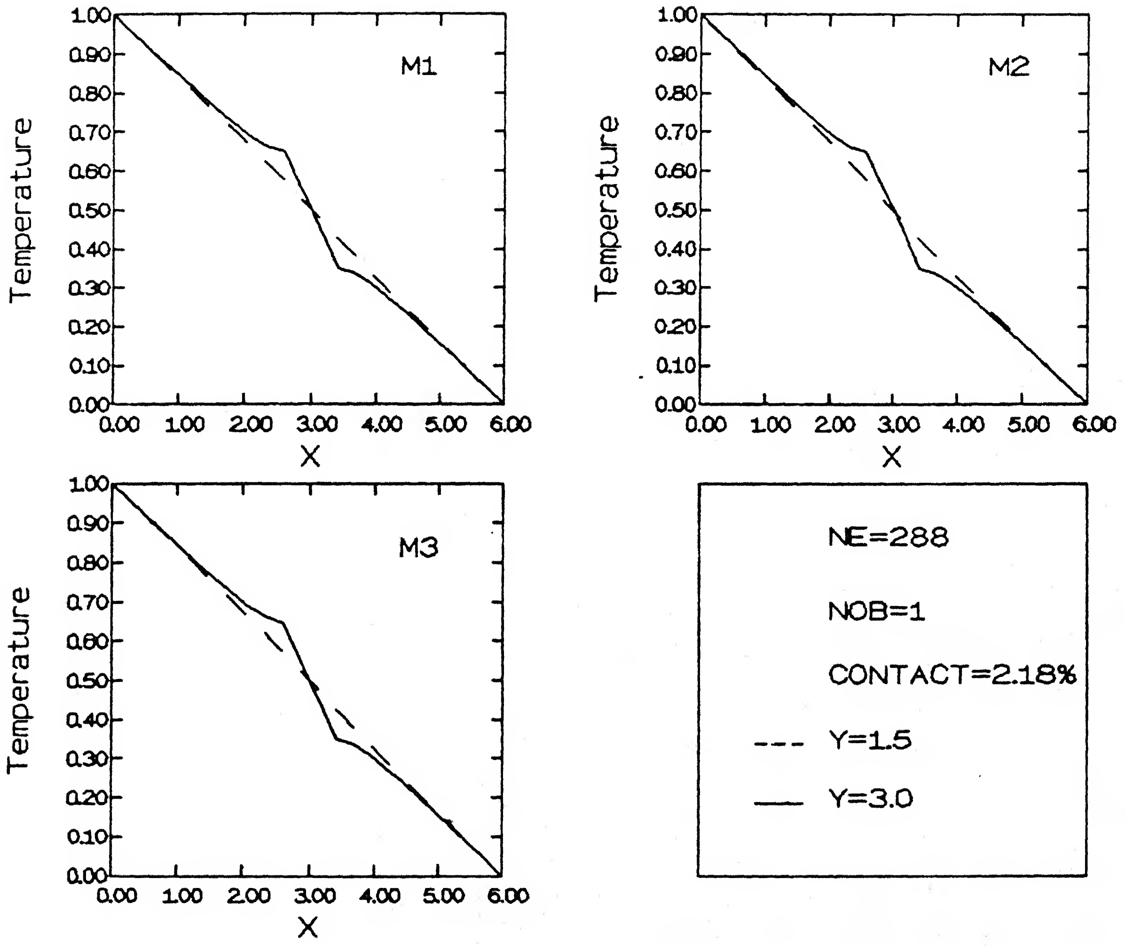


Figure 3.2 : Variation of Temperature along Heat Flow Direction
: Heat Conduction Through Heterogeneous Medium
: Number of Obstacles 1 and Number of Elements 288.

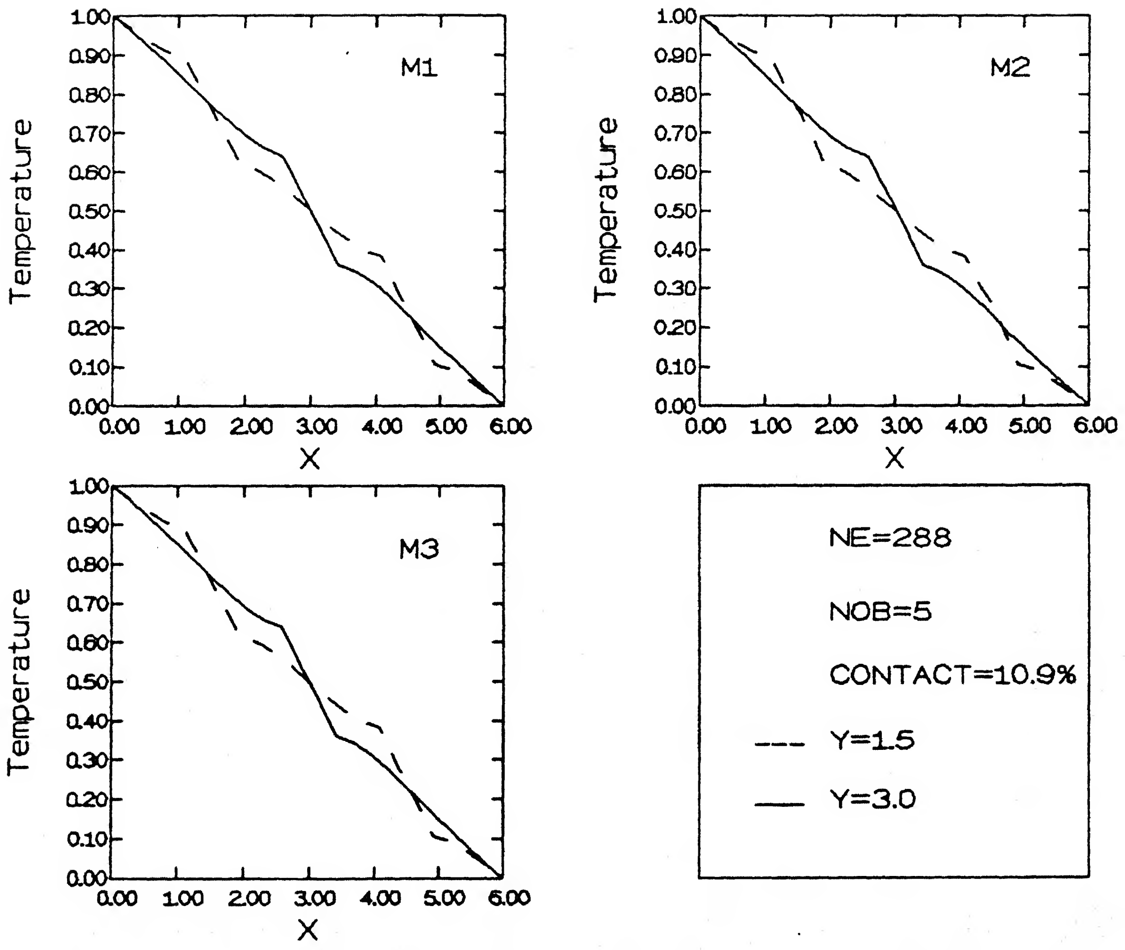


Figure 3.3 : Variation of Temperature along Heat Flow Direction
: Heat Conduction Through Heterogeneous Medium
: Number of Obstacles 5 and Number of Elements 288.

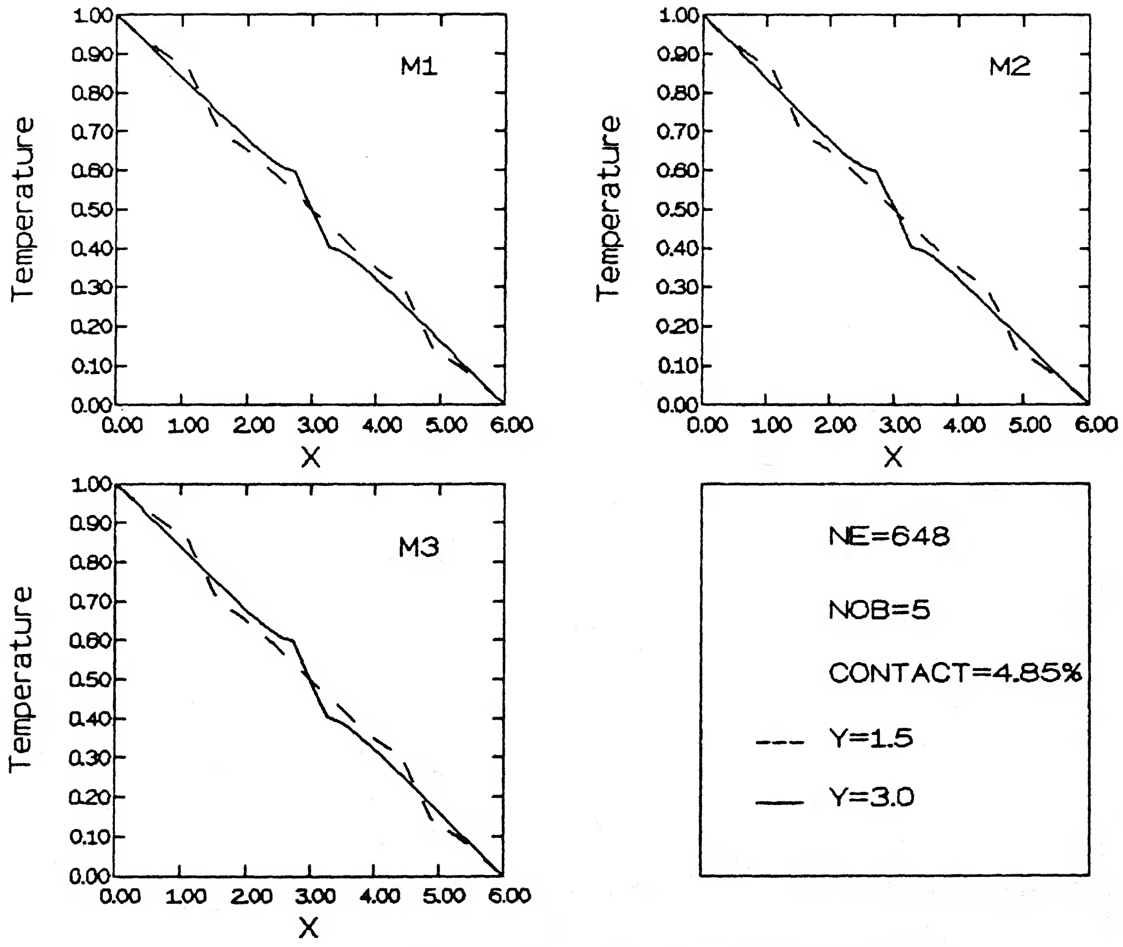


Figure 3.4 : Variation of Temperature along Heat Flow Direction
: Heat Conduction Through Heterogeneous Medium
: Number of Obstacles 5 and Number of Elements 648.

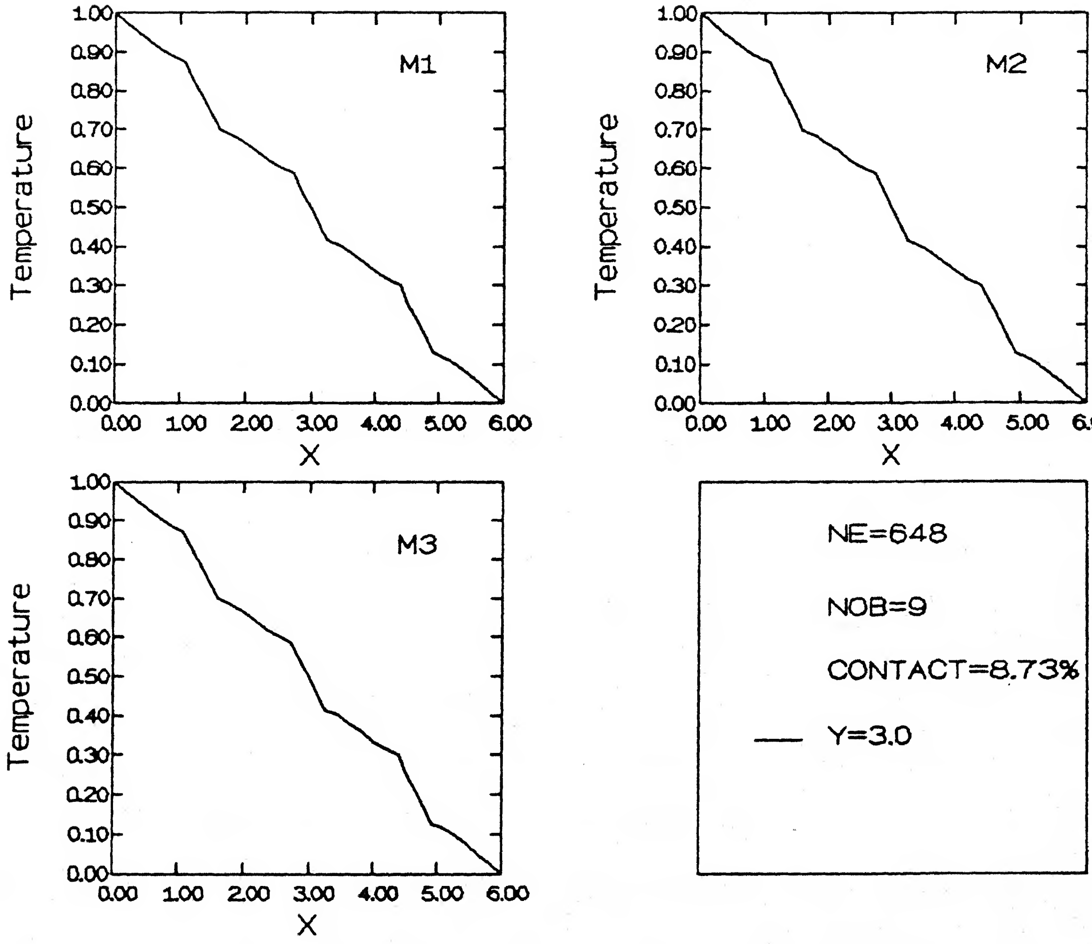


Figure 3.5 : Variation of Temperature along Heat Flow Direction
: Heat Conduction Through Heterogeneous Medium
: Number of Obstacles 9 and Number of Elements 648.

it is seen that the results obtained by three methods are exact upto the 3rd digit after the decimal. There is slight variation in the results at the 4th digit, possibly due to truncation and round off errors.

Figures 3.2, 3.3, 3.4 and 3.5 show the variation of temperature with distance x,plotted at two y locations namely y=1.5 and y=3.0. Two grids have been chosen namely a coarser grid having a number of elements NE = 288 and a finer grid with NE = 648. In the coarser grid, two types of problems considered are (a) number of obstacles, NOB = 1 (b) NOB = 5. For the finer grid the problems considered are (a) NOB = 5 (b) NOB = 9.

We can see from all the 4 graphs that wherever obstacles are present the temperature is non-uniform. The temperature profile is linear as expected in the homogeneous region.

3.4.2 CPU Time

The matrix equation is solved using three methods on the HP 9000 system. The time required for inversion of matrix is presented in Table 3.2. As shown in the Table (3.2) the three methods are compared for four different types of problems.

Table 3.2 shows that for a coarse grid (NE = 288) method M1 is more efficient than methods M2 and M3. For a finer grid (NE=648) the time required for inversion with M1 is greater as compared to method M2 but less than M3. This shows that if the size of matrix goes on increasing method M2 will be the most economical in terms of CPU time.

Method M2 takes more time for NE = 288, but there is a

Table 3.2 Comparision of CPU Time Taken by Methods $M1$, $M2$ and $M3$

CPU Time in Seconds				
Method	NE = 288	NE = 288	NE = 648	NE = 648
	NOB = 1	NOB = 5	NOB = 5	NOB = 9
$M1$	47.27	47.46	211.36	201.99
$M2$	48.66	56.24	161.08	160.97
$M3$	44.30	49.78	236.74	234.75

Table 3.3 Comparision of Memory Required by Methods $M1$, $M2$ and $M3$

Memory in kilo bytes		
Method	NE = 288	NE = 648
$M1$	1137.42	3674.75
$M2$	1031.28	3310.75
$M3$	1031.32	3310.79

considerable reduction of time for NE = 648 as compared to methods M1 and M3.

Method M3 takes less time for NE = 288 but it takes more time for NE = 648. This is due to the fact that in method M3, L&U are calculated only on nonzero diagonals of matrix. The number of nonzero diagonals remains constant for any size of grid matrix. Thus with large size of matrix L and U will be more sparse and therefore M3 will take more iterations for matrix inversion, which in turn increases CPU time.

3.4.3 Memory requirement

Table 3.3 gives the memory requirement in kilobytes for the three methods. Table shows that the memory requirement for method M1 is more as compared to M2 and M3 for both sizes of grid. Methods M2 and M3 consumes almost equal amount of memory.

3.4.4 Comparison of Method M2 and M3 with respect to Number of Iterations and Convergence Rate

Figures 3.6 and 3.7 give a clear picture of comparison of two methods M2 and M3 in terms of their convergence behaviour. The graphs shows the variation of residue versus the number of iterations. The graph shows that the method M2 converges monotonically and also takes fewer number of iteration than method M3. Method M3 almost takes twice the number of iteration compared to M2. The convergence of M3 is not monotonic.

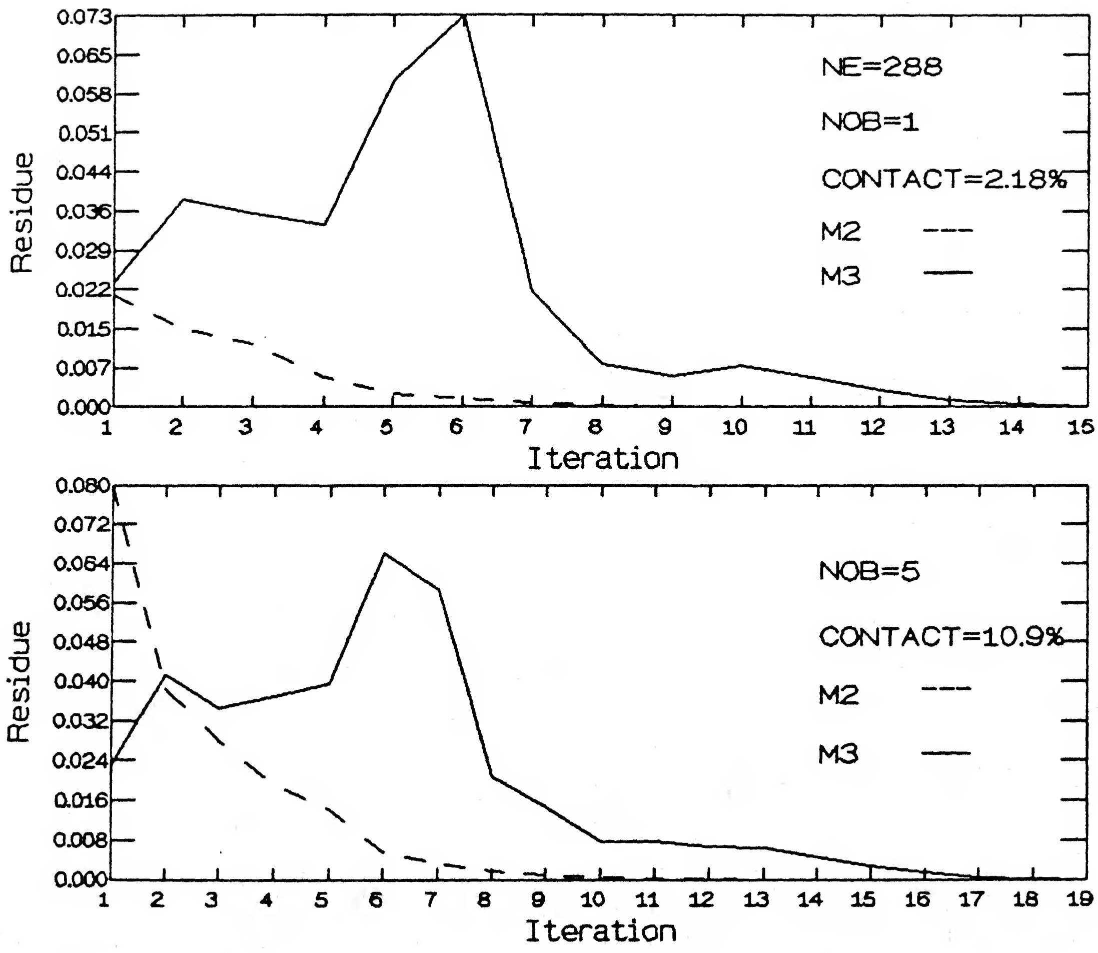


Figure 3.6 : Convergence of Residue with Number of Iterations in Methods $M2$ and $M3$
(a) : Number of Obstacles 1 and Number of Elements 288.
(b) : Number of Obstacles 5 and Number of Elements 288.

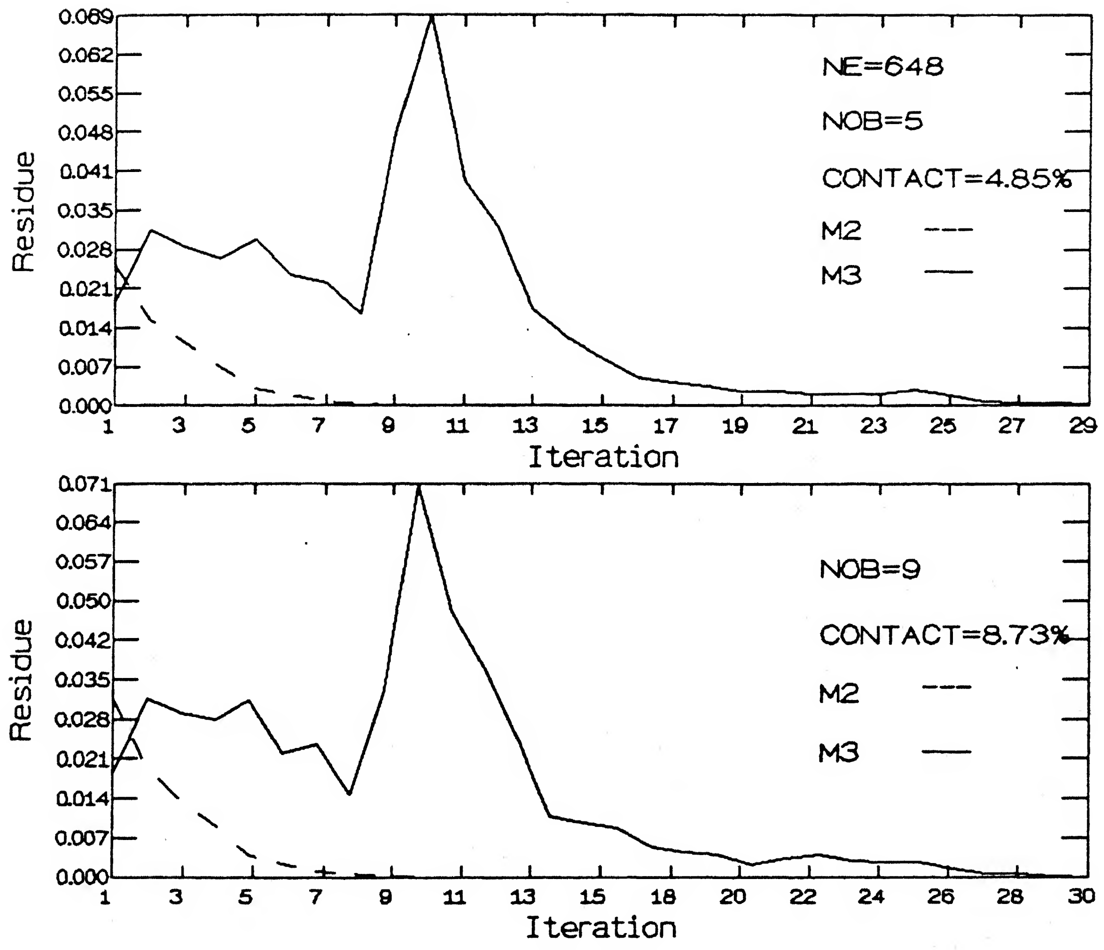


Figure 3.7 : Convergence of Residue with Number of Iterations in Methods $M2$ and $M3$
(*a*) : Number of Obstacles 5 and Number of Elements 648.
(*b*) : Number of Obstacles 9 and Number of Elements 648.

3.5 Conclusions

The most economical method is seen to be M2 as compared to M2 and M3. All the three methods show adequate accuracy. Method M2 requires less memory and also takes less time for matrix inversion. Method M2 also requires fewer number of iterations for inverting a matrix as compared to M3. Method M1 requires zero iteration as it is a direct method. The convergence of method M2 is monotonic but that of method M3 shows initially some erratic behaviour.

CHAPTER 4

APPLICATION OF PCG TO UNSTEADY NONLINEAR HEAT CONDUCTION PROBLEM

4.1 Geometric Model, Governing Differential Equation and Boundary Conditions for Unsteady Heat Conduction

The geometric model considered in this problem is similar to that used in Chapter 3. A square domain of size 6 X 6is considered. Circular inhomogeneities are distributed in the region. A temperature difference is applied in x direction while the boundaries in the y-direction are insulated.

The dimensionless form of the differential equation which governs linear unsteady heat conduction is given by,

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = \frac{\partial T}{\partial t} \tag{4.1}$$

The boundary conditions used in the present problem are given as follows:

$$\begin{aligned} x &= 0 &&; \quad T = 1 \\ x &= L &&; \quad T = 0 \\ y &= 0,\ L &&; \quad \frac{\partial T}{\partial y} = 0 \end{aligned} \tag{4.2}$$

on the fibre surface $\frac{\partial T}{\partial n} = 0.$

The initial condition corresponds to the cold state,

$$t = 0 \qquad T = 0 \tag{4.3}$$

The finite element formulation for Equations 4.1 and 4.2 is given in Appendix-B. Integration in time continues till average heat flux on the plane x = L stabilizes to within 0.01%.

This problem is solved by three methods M1, M2 and M3 and are compared among themselves with respect to CPU time and memory requirement. Three separate problems considered are the following:

(a) NE = 288 ; NOB = 0

(b) NE = 648 ; NOB = 5

(c) NE = 648 ; NOB = 9.

4.2 Comparison with Respect to CPU Time

The CPU time comparison for the three methods for the three different problems is given in Table 4.1. Table 4.1 shows that method M1 is the most efficient of the three methods. This is due to the fact that in this method the matrix is inverted for the first time step and then the inverted matrix is stored. For the next time step it just multiplies the right hand side vector by the inverted matrix. While in methods M2 and M3 repeated matrix inversion is carried out for each time step. This leads to an increase in CPU time. Table 4.1 also shows that the method M2 is more efficient than method M3.

4.3 Comparison with respect to Memory requirement

A comparison of the three methods with respect to memory requirement is presented in Table 4.2. Table 4.2 shows that the memory requirement of M1 is greater than method M2 and Method M3. Methods M2 and M3 consume almost equal memory.

In this problem method M2 has been used in a slightly modified form. Instead of supplying L and U from the zero obstacle problem, L and U are calculated for the first time step and then they are used for subsequent time steps. This way of supplying L and U is more efficient than supplying L and U from

Table 4.1 Comparision of CPU Time Taken by Methods $M1$, $M2$ and $M3$

CPU Time in Seconds			
Method	NE = 288	NE = 648	NE = 648
	NOB = 0	NOB = 5	NOB = 9
$M1$	77.04	316.44	323.88
$M2$	204.12	792.00	1074.24
$M3$	316.20	1545.12	1653.12

Table 4.2 Comparision of Memory Required by Methods $M1$, $M2$ and $M3$

Memory in kilo bytes		
Method	NE = 288	NE = 648
$M1$	1193.89	3800.92
$M2$	1090.27	3442.41
$M3$	1090.29	3442.43

the zero obstacle problem. Storing L and U matrices also leads to an increase in memory size.

4.4 Comparison of M2 and M3

Method M2 is clearly better than M3 and is selected for further computation . A comparison of methods M1 and M2 for a nonlinear problem is considered in the section 4.6.

4.5 Results

Figure 4.1 and Figure 4.2 shows the results for case (a) (NE = 288, NOB = 0). Figure 4.1 shows the heat flux as a function of time. At the beginning of the simulation **EQCI** is much higher than **EQCO**. This is due to the fact that the heat flux has not reached to the right hand boundary. With time **EQCI** reduces and **EQCO** increases. When steady state is reached **EQCI** will be equal to **EQCO**. Figure 4.3 and Figure 4.4 are results of the problem NE = 648, NOB = 5. The Figure 4.3 shows that the heat flux is reduced. This is due to the presence of the obstacles, which increases resistance to the flow of heat. Figure 4.5 and 4.6 are the results of the problem NE = 648 and NOB = 9. We see that as the number of obstacles increases the heat flux is reduced.

4.6 Governing Equation and Boundary Condition for a Nonlinear Heat Conduction

In this section the nonlinear unsteady heat conduction problem is solved by two methods namely M1 and M2. The two methods are compared with respect to CPU time and memory

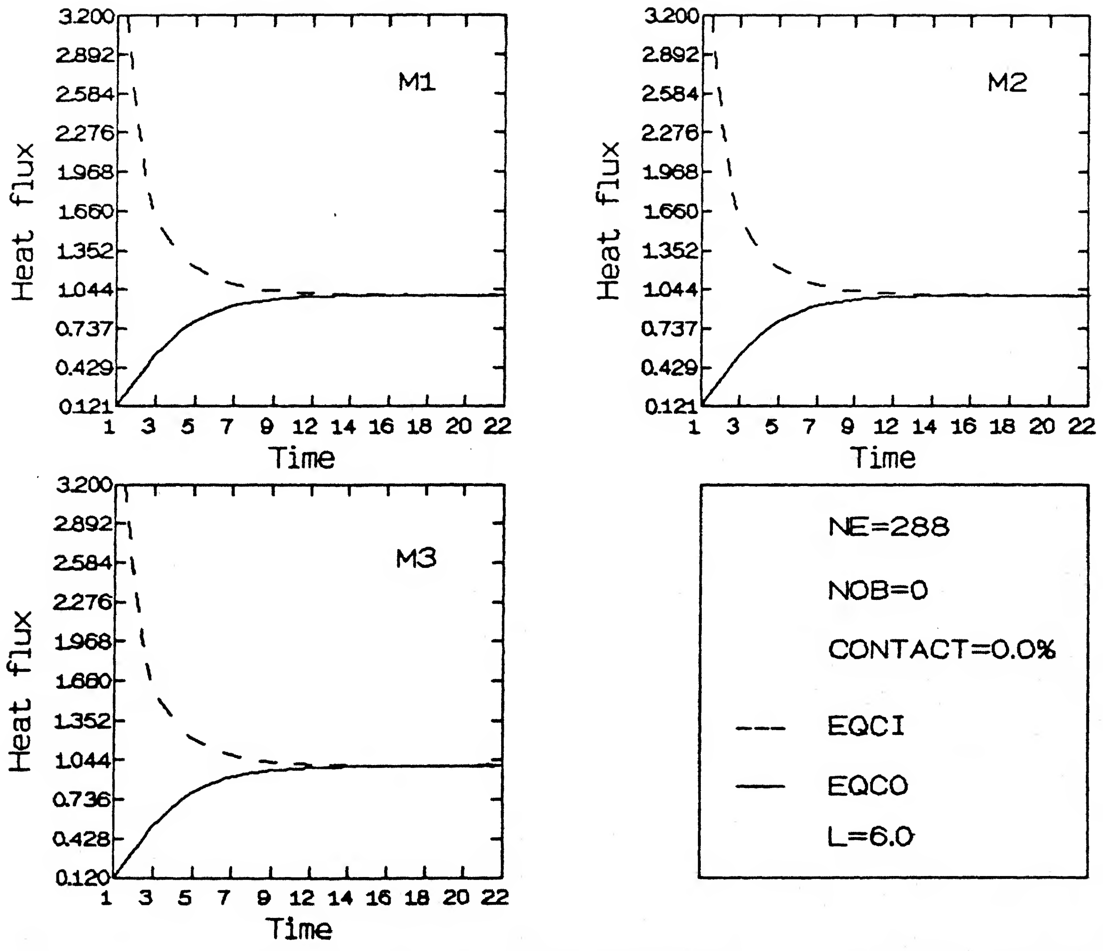


Figure 4.1 : Temporal Variation of Heat Flux at Entrance and Exit Planes
: Transient Heat Conduction Through Homogeneous Medium.

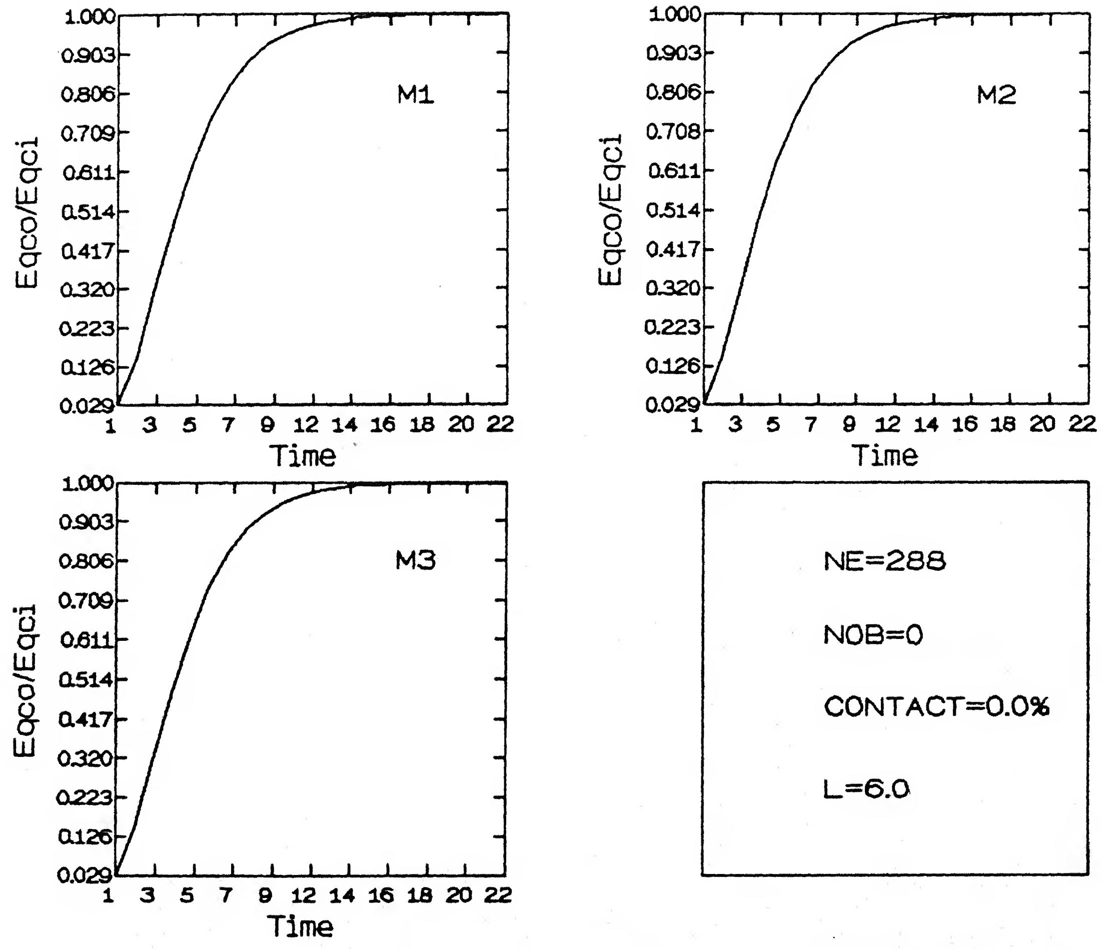


Figure 4.2 : Temporal Variation of Ratio of Entrance to Exit Heat Fluxes
: Transient Heat Conduction Through Homogeneous Medium.

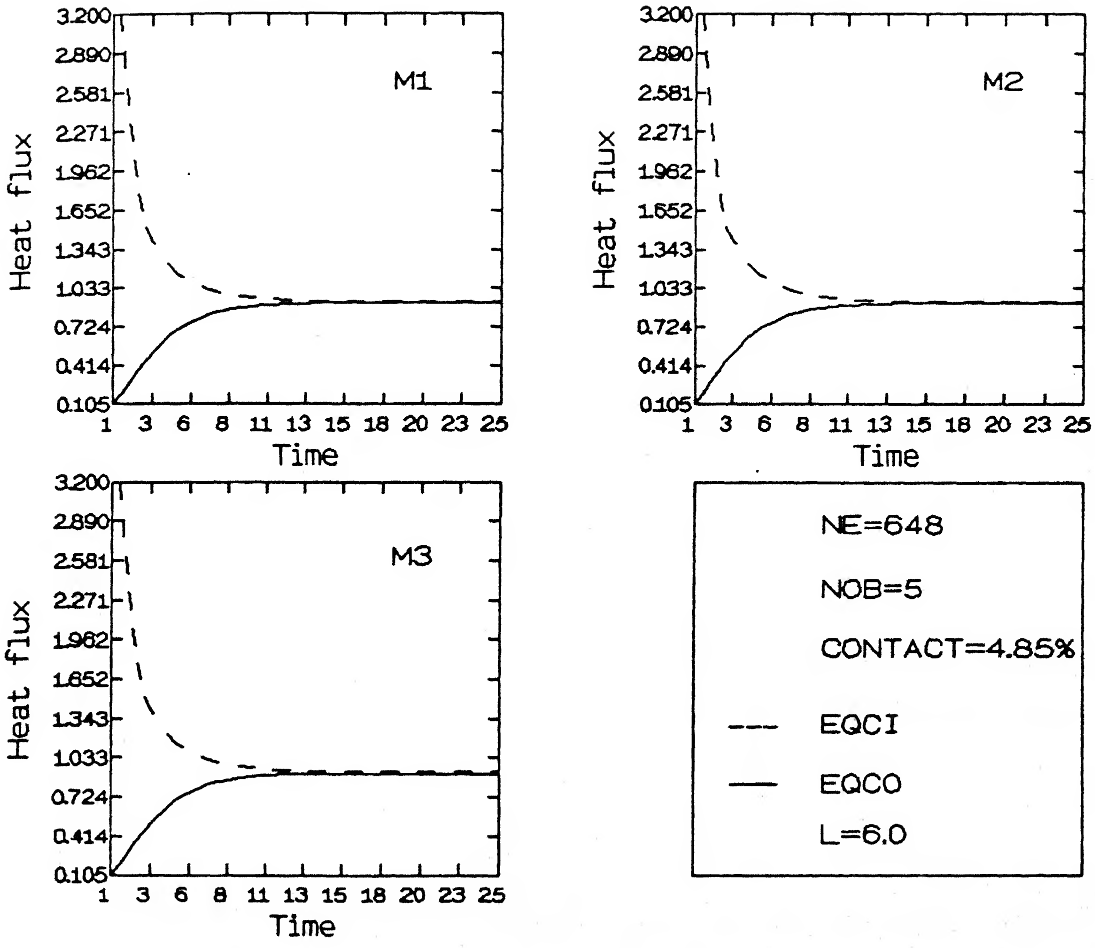


Figure 4.3 : Temporal Variation of Heat Flux at Entrance and Exit Planes
: Transient Heat Conduction Through Heterogeneous Medium.
: Number of Obstacles 5 and Number of Elements 648 .

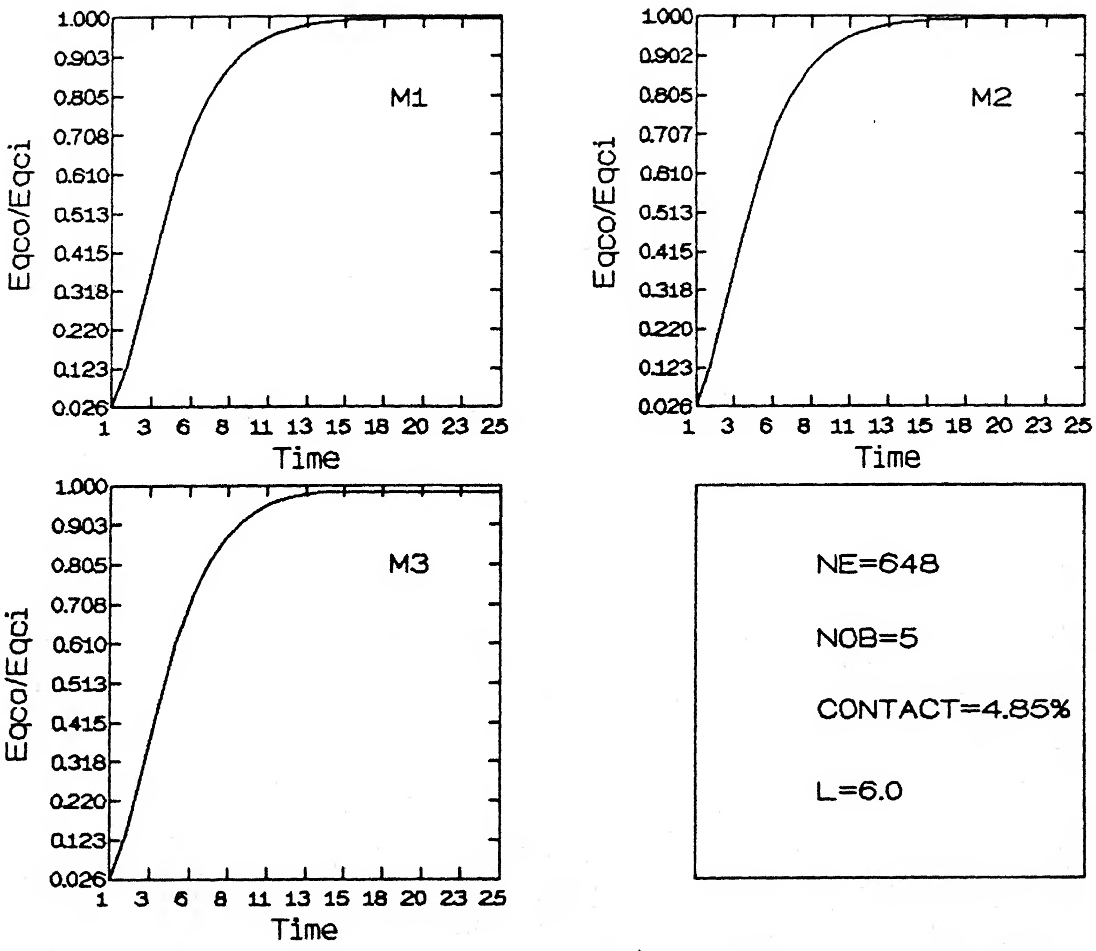


Figure 4.4 : Temporal Variation of Ratio of Entrance to Exit Heat Fluxes
: Transient Heat Conduction Through Heterogeneous Medium.
: Number of Obstacles 5 and Number of Elements 648 .

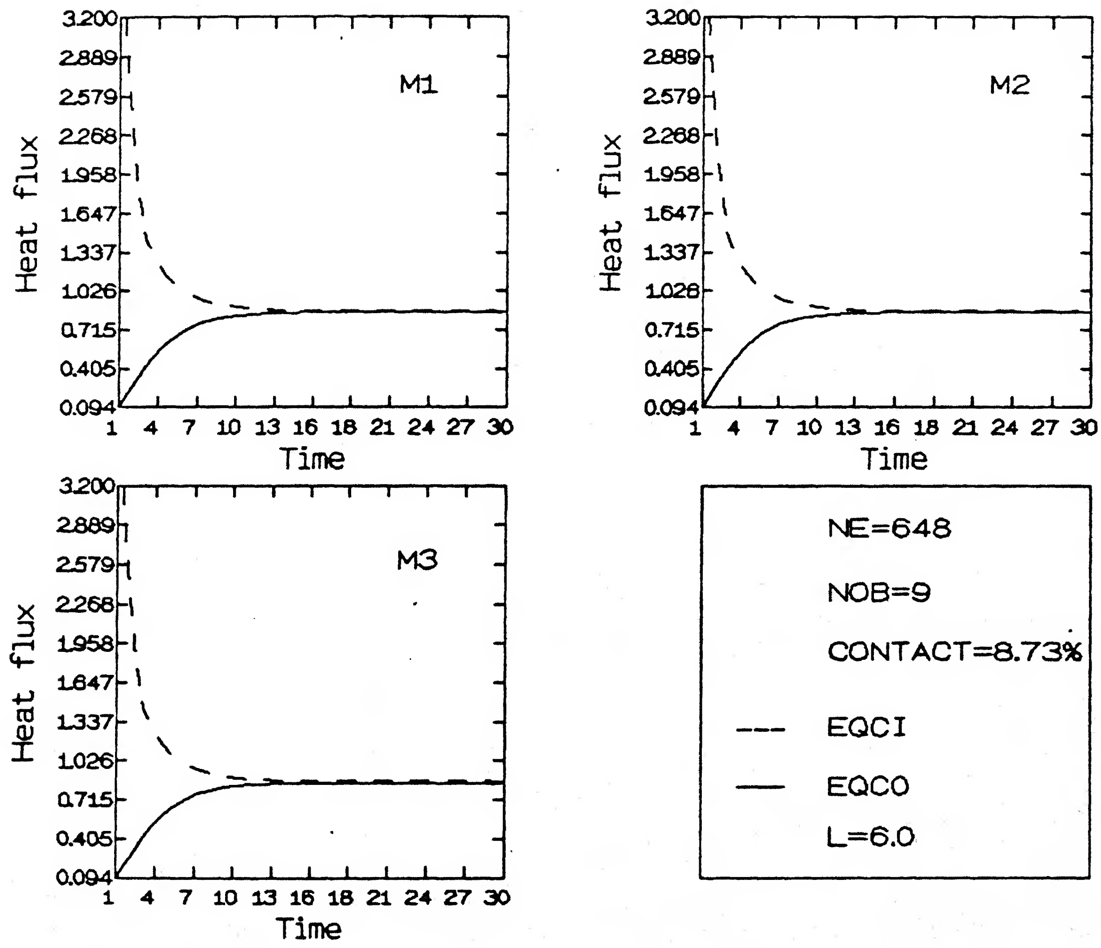


Figure 4.5 : Temporal Variation of Heat Flux at Entrance and Exit Planes
: Transient Heat Conduction Through Heterogeneous Medium.
: Number of Obstacles 9 and Number of Elements 648 .

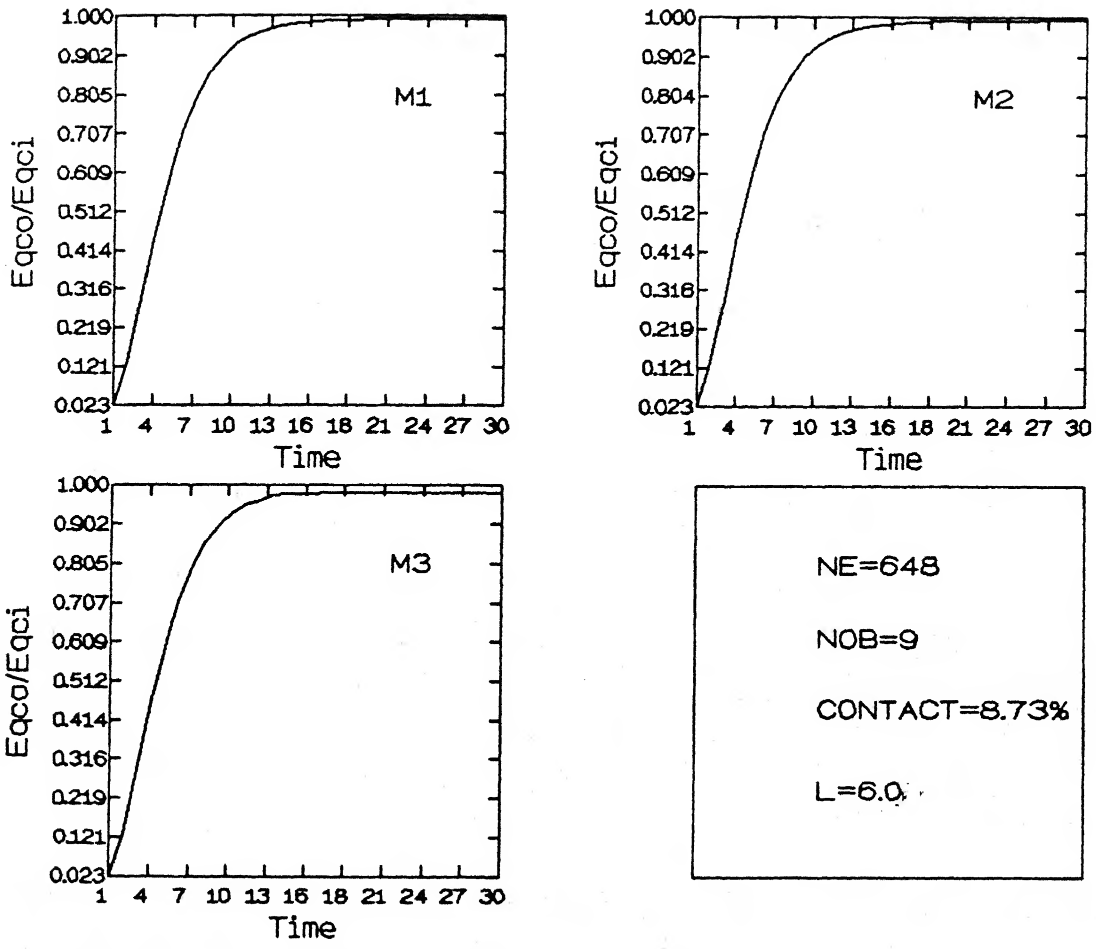


Figure 4.6 : Temporal Variation of Ratio of Entrance to Exit Heat Fluxes
: Transient Heat Conduction Through Heterogeneous Medium.
: Number of Obstacles 9 and Number of Elements 648 .

requirement.

The equation which governs the nonlinear heat conduction problem in nondimensional form is,

$$\frac{\partial}{\partial x}\left(K(T)\frac{\partial t}{\partial x}\right) + \frac{\partial}{\partial y}\left(K(T)\frac{\partial T}{\partial y}\right) = \frac{\partial T}{\partial t} \qquad (4.4)$$

The boundary conditions chosen for the present study are given as follows:

$$\begin{array}{lll} x = 0 & ; & T = 1 \\ x = L & ; & T = 0 \\ y = 0,\ L & ; & \dfrac{\partial T}{\partial y} = 0 \end{array} \qquad (4.5)$$

for the obstacles $\frac{\partial T}{\partial n} = 0$.

The initial condition corresponds to the cold state,

$$t = 0 \quad ; \quad T = 0 \qquad (4.6)$$

Thermal conductivity is taken to vary as

$$k = 1 - \beta T \qquad (4.7)$$

This is valid for metals such as copper.

The value of $\beta = 0.4$ has been used in the present work. In the finite element implementation $\dot{T}$ in Equation 4.7 is interpreted as $T = (T_1 + T_2 + T_3 + T_4 + T_5 + T_6)/\ 6$ where $T_1 \ldots\ldots\ldots T_6$ are temperatures at each node of the triangular element.

The element level matrix is obtained by weighted integration of the governing equation over an element (Appendix-B).

$$\left[\int_{\Omega^e} N_i . N_j \, dx\, dy\right] \frac{dT_j}{\partial t} + \int_{\Omega^e} K \left[\frac{\partial N_i}{\partial x} . \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} . \frac{\partial N_j}{\partial y}\right] dx\, dy\ \{T_j\}=0$$

(4.8)

The right hand side of (4.8) is zero due to boundary conditions (4.5). An implicit scheme is used to solve Equation

4.8. The marching algorithm is of the form $T_i^{n+1} = C_{ij} T_j^n$,

where

$$C_{ij} = \left[A_{ij} + \Delta t\ B_{ij}\right]^{-1} A_{ij}$$

$$A_{ij} = \int_{\Omega^e} N_i N_j \, dx\, dy$$

$$B_{ij} = \int_{\Omega^e} K \left[\frac{\partial N_i}{\partial x} \cdot \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \cdot \frac{\partial N_j}{\partial y}\right] dx\, dy.$$

4.7 Comparison with respect to CPU Time

Table 4.3 represents comparison of two methods M1 and M2 with respect to time. Table 4.3 shows that the CPU time for method M2 is lesser than that of method M1. With a finer grid method M1 almost requires 3.5 times the CPU-time of method M2. This is due to the fact that for every new time level the matrix changes and in M1 one has to completely invert the matrix. Method M2 takes more time for LU decomposition at the first time step but after that the Conjugate Gradient computation is fast as it has to do fewer number of iterations.

4.8 Comparison with respect to Memory requirement

Table 4.4 shows the comparison of memory requirement for methods M1 and M2. It is seen that method M1 requires more memory than method M2. The difference in memory requirement of methods M1 and M2 increases as the fineness of the grid decreases.

Table 4.3 Comparision of CPU Time Taken by Methods $M1$ and $M2$

CPU Time in Seconds			
Method	NE = 288	NE = 512	NE = 512
	NOB = 0	NOB = 5	NOB = 9
$M1$	2021.04	20460.15	22245.30
$M2$	1821.84	5438.64	6271.32

Table 4.4 Comparision of Memory Required by Methods $M1$ and $M2$

Memory in kilo bytes		
Method	NE = 288	NE = 512
$M1$	1457.30	3307.80
$M2$	1108.92	2491.48

4.9 Comparison of M1 and M2

The above two comparisons shows that the method M2 is more economical than method M1 with regard to CPU time and memory requirement. Method M2 has been applied to more complex problems in the Chapters 5 and 6.

4.10 Results

Figures 4.7, 4.8 and 4.9 show a summary of results obtained in the present work. Two plots are shown namely,

(a) Heat flux versus time

(b) Ratio of heat flux at outlet to heat flux at inlet versus time.

Figures 4.7, 4.8 and 4.9 shows that initially the heat flux at x=0 is quite high as compared to heat flux at x=L. There is a sharp decrease in the heat flux at inlet boundary with time. After some time the average heat flux on the left boundary becomes equal to heat flux on the right boundary and the region reaches steady state. The graphs also shows that the steady state equivalent thermal conductivity of the medium goes on decreasing as the number of obstacles increases.

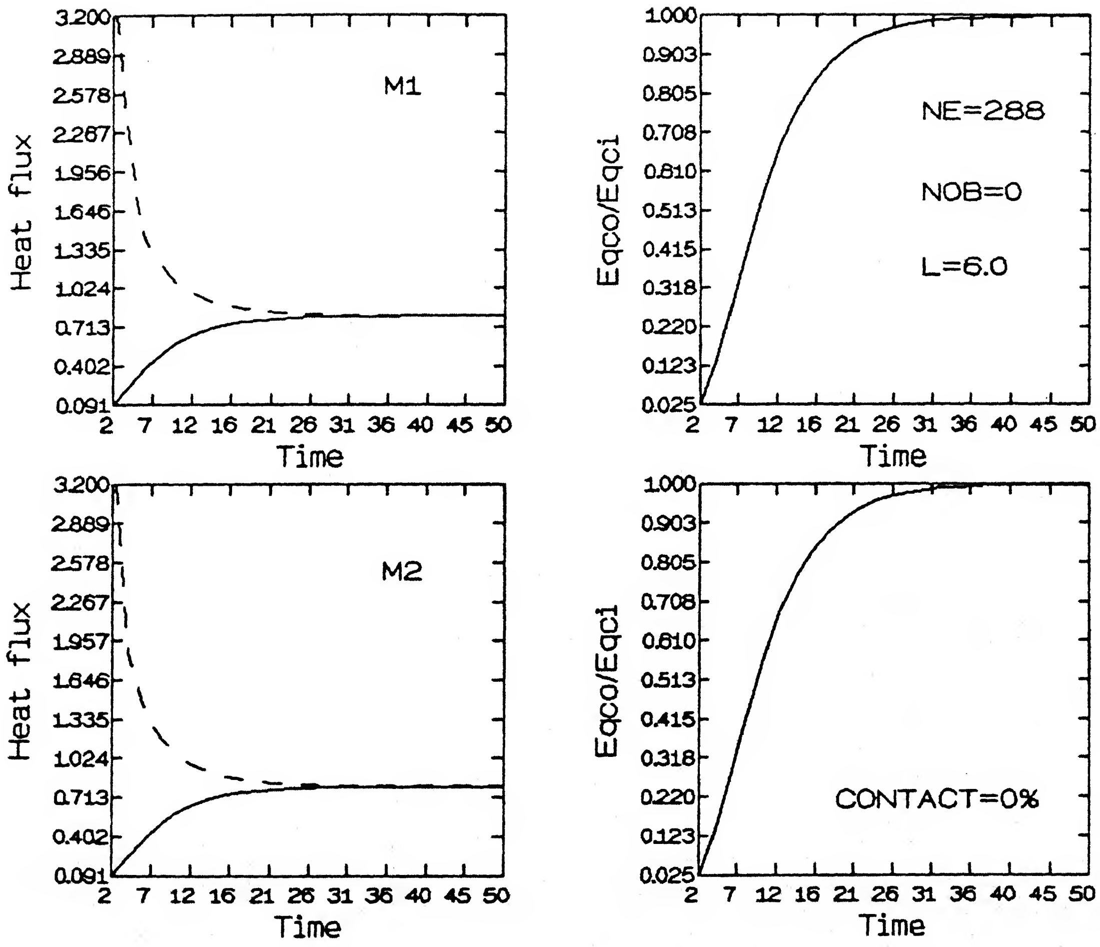


Figure 4.7 *a* & *b* : Temporal Variation of Heat Flux at Entrance and Exit Planes

c &*d* : Temporal Variation of Ratio of Entrance to Exit Heat Fluxes

: Transient Heat Conduction Through Homogenous Medium.

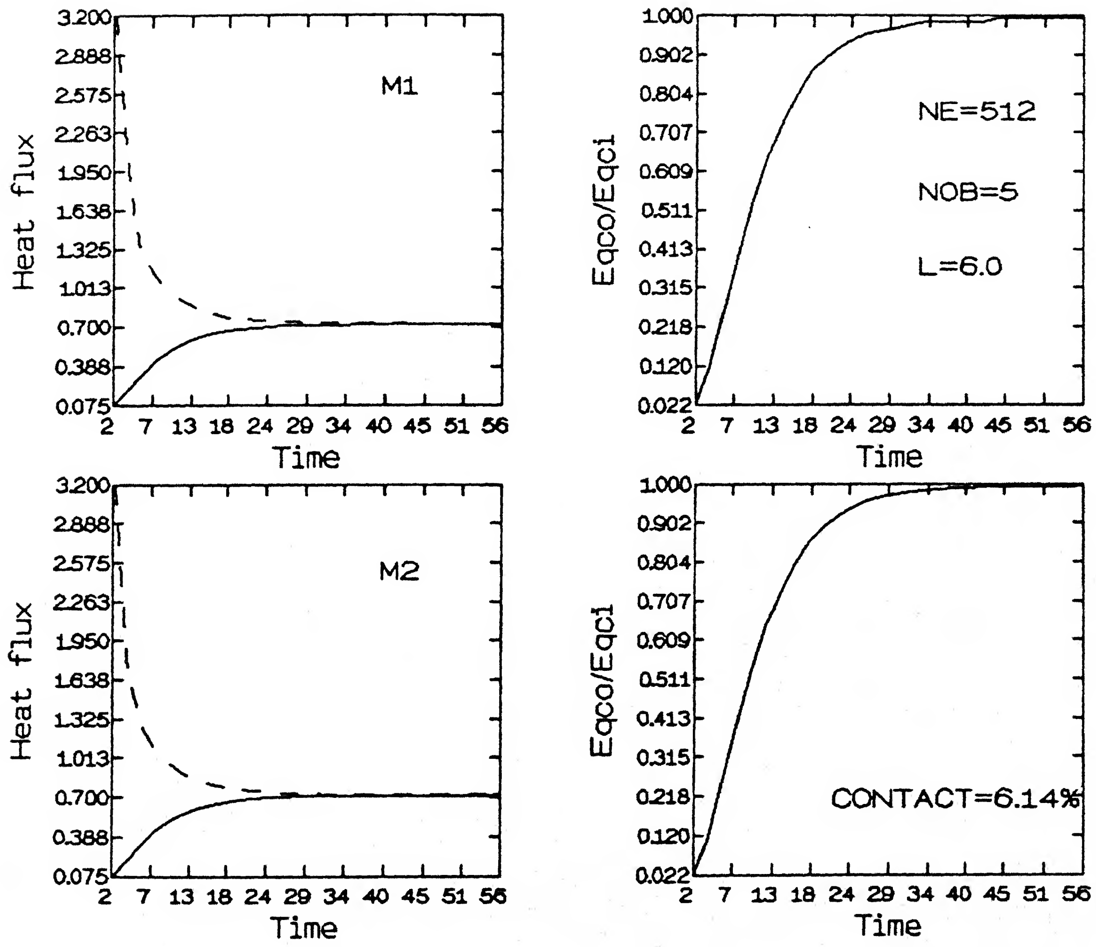


Figure 4.8 *a* & *b* : Temporal Variation of Heat Flux at Entrance and Exit Planes
c &*d* : Temporal Variation of Ratio of Entrance to Exit Heat Fluxes
: Transient Heat Conduction Through Heterogeneous Medium.
: Number of Obstacles 5 and Number of Elements 512 .

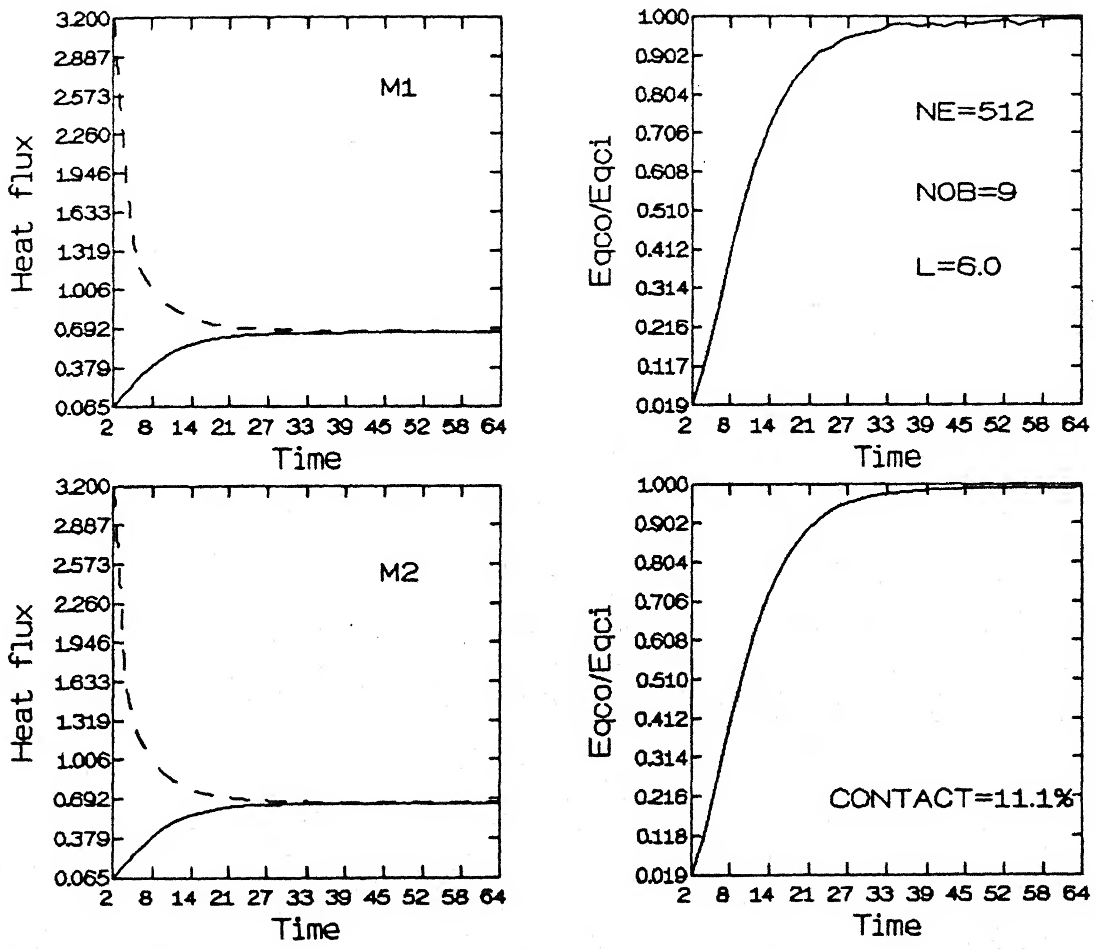


Figure 4.9 *a* & *b* : Temporal Variation of Heat Flux at Entrance and Exit Planes
c &*d* : Temporal Variation of Ratio of Entrance to Exit Heat Fluxes
: Transient Heat Conduction Through Heterogeneous Medium.
: Number of Obstacles 9 and Number of Elements 512 .

CHAPTER 5
SIMULATION OF OIL FLOW IN ROCK FRACTURE

5.1 Introduction

In this chapter flow of a non-Newtonian fluid in a single rock fracture is studied. The relevant application is extraction of hydrocarbon oil from fractured rocks. Each fracture is modeled as a parallel channel with a distribution of circular contact areas where the fracture aperture is zero. A pressure drop is applied across the fracture. The flow rate is calculated by solving the depth averaged Stokes equations.

Results are presented for two types of fluids. First is a powerlaw fluid and the second is a viscoelastic fluid. The flow behaviour for both the fluids is given by Herschel-Bulkley model. The model covers the flow of homogeneous fluids with a yield stress as well as those which display power law behaviour. The effect of contact areas on the flow rate has been studied here.

5.2 Geometric Model

The rock fracture has been modeled in this work as consisting of parallel planes with a distribution of contact areas. This is shown in Figure 5.1. It is assumed that the variation of aperture between contact areas is unimportant, as far as the pressure drop problem is concerned. This implicitly states that contact areas are the primary cause of difference between fracture modeled as parallel plates and laboratory experiments. The parallel plate model of a fracture with a distribution of circular contact areas

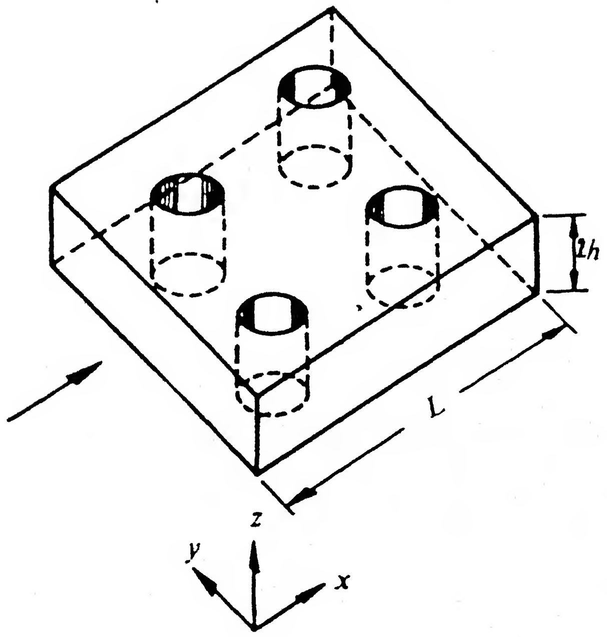


Figure 5.1 : Model of Rock Fracture with Contact Areas is equivalent to the idealization of soil as a homogeneous isotropic porous medium. The contact areas are assumed to be circular in shape.

The flow is two dimensional and the rectangular domain has a nominal size of 1m x 1m x 2mm, the smallest dimension being the aperture of the fracture. In a Cartesian plane x=0 and x = 1m are the inflow and outflow planes respectively. The upper and lower edges of the domain (y = 0 and y = 1m) are symmetry planes and hence do not permit a net normal velocity. The goal of the study is to determine the extent of oil recovery when the fracture is subjected to a prescribed pressure difference.

5.3 Formulation

The formulation of the governing differential equations is presented here in dimensional form. The full equations governing fluid flow in the fracture are the Navier-Stokes equations.

$$\rho_o \left[\frac{\partial u_o}{\partial t} + u_o.\nabla u_o \right] = \Psi + \nabla.\sigma \qquad (5.1)$$

$$\frac{\partial \rho_o}{\partial t} + \nabla.\rho_o u_o = 0 \qquad (5.2)$$

Since the characteristic velocity of oil is quite small, Reynolds number is also expected to be very small. Under these circumstances it is assumed that the inertial terms $u_o.\nabla u_o$ are uniformly small and hence can be dropped from the momentum equation (*Schlichling* 1979). Further the fracture aperture is also a small quantity in relation to its length, i.e. $\frac{h}{L} << 1$ where L is the linear fracture dimension. Since the velocity component w, normal to the fracture plane scales as $\frac{w}{u_o} = O(\frac{h}{L})$, we assume that w is identically zero in the fracture. This is equivalent to saying that the pressure is uniform across fracture height. With these simplifications, the flow equations (5.1) and (5.2) reduce to the following form:

$$\nabla.\sigma = 0 \qquad (5.3)$$

$$\nabla.u_o = - \frac{1}{\rho_o} \frac{\partial \rho_o}{\partial t} \qquad (5.4)$$

For a two dimensional region these equations can be written in a expandedformas,

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} = 0 \quad (5.5)$$

$$\frac{\partial \sigma_{yx}}{\partial x} + \frac{\partial \sigma_{zz}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} = 0 (5.6)$$

The constitutive equation for power law fluid is given by

$$\sigma_{ij} = -p\ \delta_{ij} + k\ (2\ \dot{\varepsilon}_{ij})^n \qquad (5.7)$$

$$\dot{\varepsilon}_{ij} = \frac{1}{2} \left[\frac{\partial u_{oi}}{\partial x_j} + \frac{\partial u_{oj}}{\partial x_i} \right] \qquad (5.8)$$

Substituting these constitutive relations into (5.5) and (5.6) we get the applicable x momentum and y momentum equations as follows:

$$\frac{\partial}{\partial z} \left\{ K_o \left[\frac{\partial u_o}{\partial z} \right]^n \right\} = \frac{\partial p_o}{\partial x} \qquad (5.9)$$

$$\frac{\partial}{\partial z} \left\{ K_o \left[\frac{\partial v_o}{\partial z} \right]^n \right\} = \frac{\partial p_o}{\partial y} \qquad (5.10)$$

since p is only a function of x and y, each side of equations (5.9) and (5.10) are constant. The continuity equation in two dimension form is obtained from equation (5.4) as,

$$\frac{\partial u_o}{\partial x} + \frac{\partial u_o}{\partial y} = -\frac{1}{\rho_o}\left[\frac{\partial \rho_o}{\partial p_o} \cdot \frac{\partial p_o}{\partial t}\right] = \xi_o \cdot \frac{\partial p_o}{\partial t} \qquad (5.11)$$

where ξ_o is the compressibility of the fluid.

Equations (5.9) and (5.10) can be solved subject to the boundary conditions

$$z = h \quad ; \quad u_o = 0,\ v_o = 0$$

$$z = 0 \quad ; \quad \frac{\partial u_o}{\partial z} = 0,\ \frac{\partial v_o}{\partial z} = 0 \qquad (5.12)$$

Hence,

$$u_o = \left[\frac{n}{n+1}\right]\left[\frac{1}{K_o} \cdot \frac{\partial p_o}{\partial x}\right]^{1/n}\left[z^{(n+1)/n} - h^{(n+1)/n}\right] \qquad (5.13)$$

$$v_o = \left[\frac{n}{n+1}\right]\left[\frac{1}{K_o} \cdot \frac{\partial p_o}{\partial y}\right]^{1/n}\left[z^{(n+1)/n} - h^{(n+1)/n}\right] \qquad (5.14)$$

To eliminate the z co-ordinate these equations (5.13) and (5.14) are depth averaged over the fracture aperture. These average velocities are

$$\bar{u}_o = \alpha \left[\frac{\partial p_o}{\partial x}\right]^{1/n} \qquad (5.15)$$

$$\bar{v}_o = \alpha \left[\frac{\partial p_o}{\partial y}\right]^{1/n} \qquad (5.16)$$

$$\alpha = \left[\frac{-n}{2n+1}\right]\left[h^{(n+1)/n}\right]\left[\frac{1}{k}\right]^{1/n} \qquad (5.17)$$

$$\frac{\partial \bar{u}_o}{\partial x} + \frac{\partial \bar{v}_o}{\partial x} = \xi_o \cdot \frac{\partial p_o}{\partial t} \qquad (5.18)$$

Substituting these average velocities (5.15) and (5.16) into equation (5.18) we get finally only one equation in pressure:

$$\frac{\partial}{\partial x}\left[\frac{\partial p_o}{\partial x}\right]^{1/n} + \frac{\partial}{\partial y}\left[\frac{\partial p_o}{\partial y}\right]^{1/n} = \frac{\xi_o}{\alpha}\cdot\frac{\partial p_o}{\partial t} \qquad (5.19)$$

The values of various fluid parameters in equation (5.19) are given in Table 5.1 (*Al.fariss and Pinder* 1987). The boundary conditions for equation (5.19) are as follows:

$$\begin{aligned} x &= 0 & p_o &= 1.793 \text{ MPa} \\ x &= L & p_o &= 1.31 \text{ MPa} \qquad (5.20) \\ y &= 0 \ : L & \frac{\partial p_o}{\partial y} &= 0 \end{aligned}$$

The initial conditions are taken as

$t = 0$; p_o = 1.31 MPa, so that the fluid is quiescent.

To prevent numerical oscillations, a mild gradient in pressure is however used at $t = 0$.

The flow rate of great practical value is calculated at the inflow and the outflow planes of the fracture. The flow rate on any of these boundaries is given by the equation

$$\int_0^L \left[\frac{n}{2n+1}\right]\left[\frac{1}{K_o}\cdot\frac{\partial p_o}{\partial x}\right]^{1/n}\left[h^{(2n+1)/n}\right] dy \qquad (5.21)$$

The integral is evaluated using Simpson's rule using as many as nodes. The quantity $\frac{\partial p_o}{\partial x}$ is evaluated using a three point forward difference formula at the inflow plane and by three point backward difference formula at the outflow plane.

Equation (5.19) subject to boundary conditions (5.20) have been solved by the Galerkin finite element method. The fracture

Table 5.1 Properties of a Power-Law Fluid with no Yield Stress

Compressibility of oil (ζ_o)	0.03447 Pa^{-1}
Power-law Index (n)	0.6
Consistency Index (K_o)	1.5 $Pa.s^{0.6}$
Inlet Pressure (P_1)	1.793 MPa
Exit Pressure (P_2)	1.31 Mpa

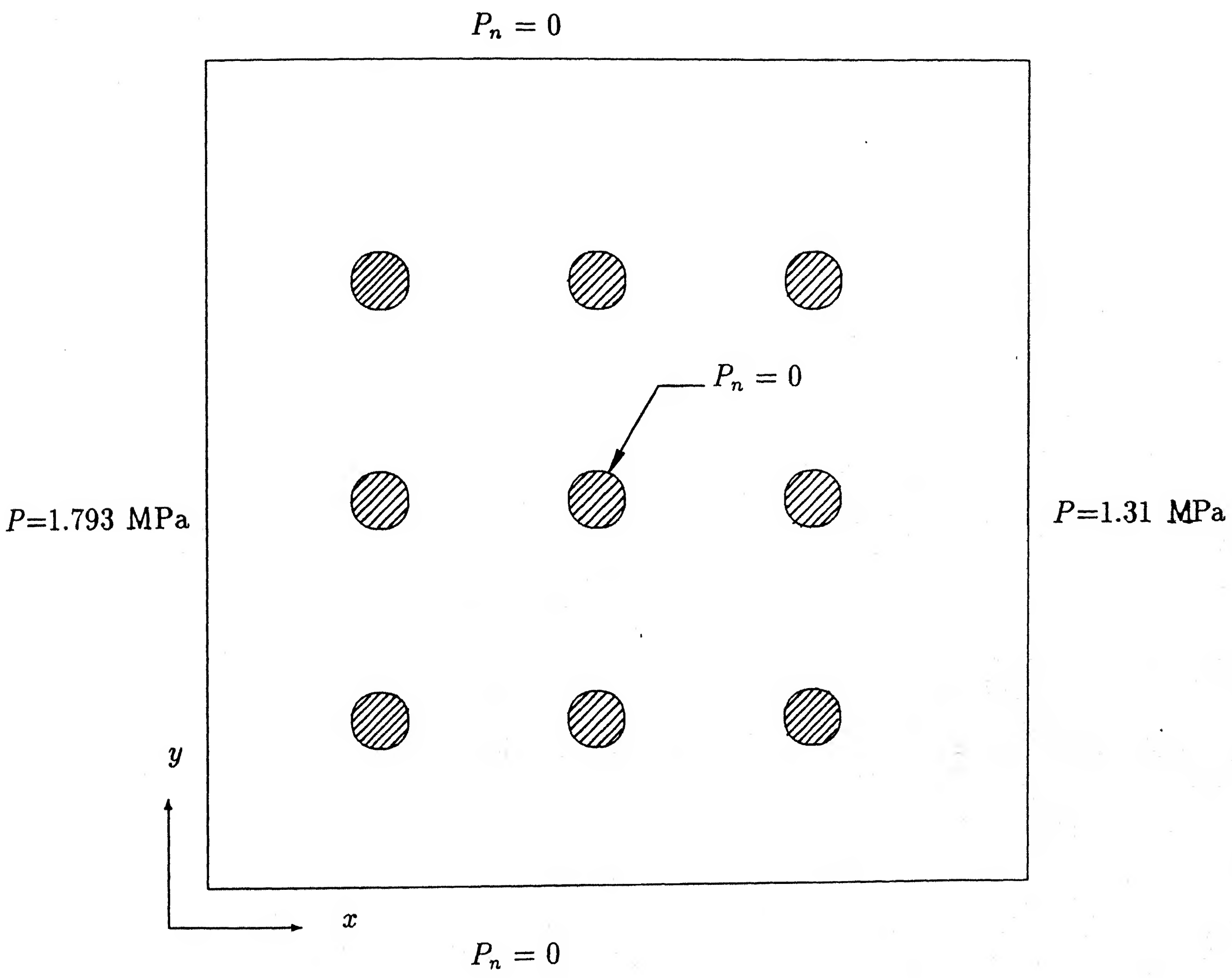


Figure 5.2 : Fracture Plane and Coordinate System

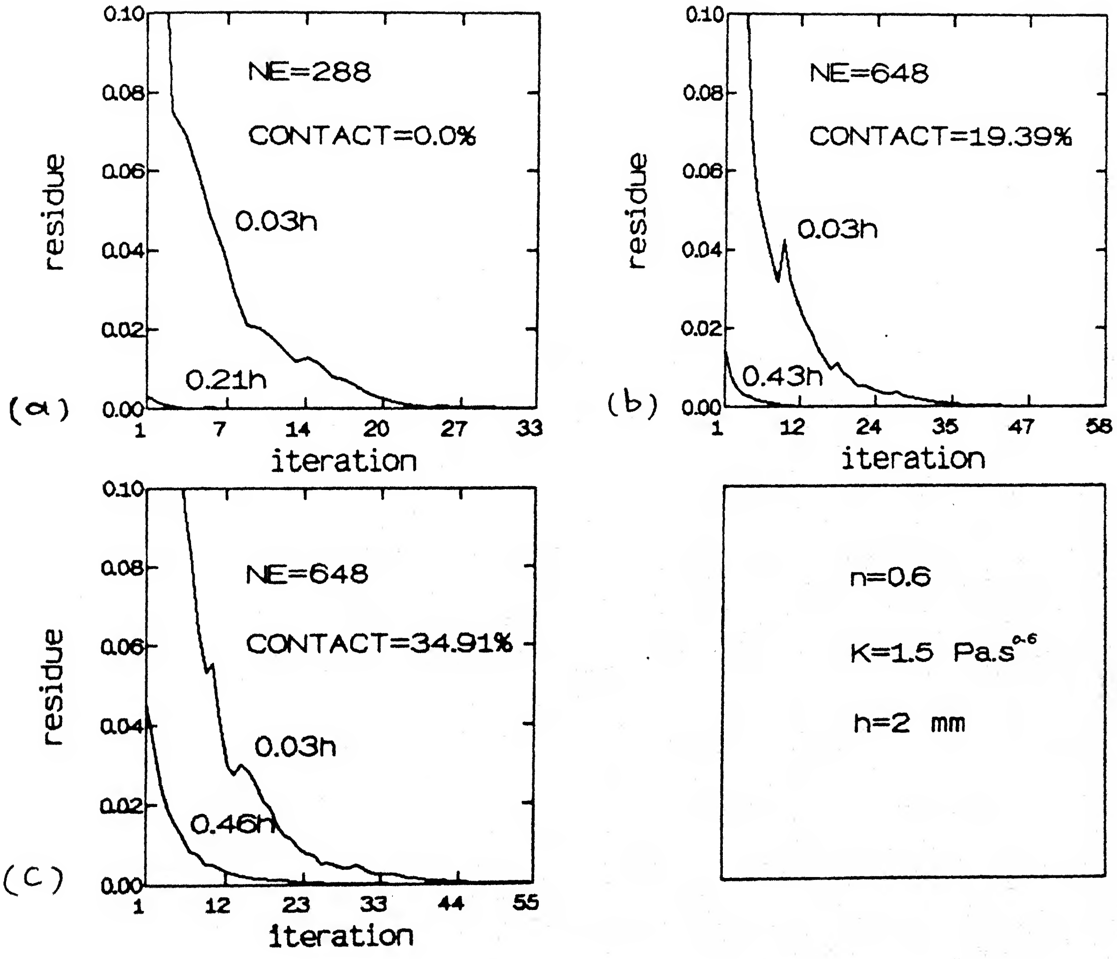


Figure 5.3 : Convergence of Residue with Number of Iterations in Method $M2$

plane with a coordinate system is shown in Figure 5.2. The system of algebraic equations has been inverted by the Preconditioned Conjugate Gradient Method (M2) developed in the present work.

5.4 Results

Results for power law fluid with no internal shear stress is considered first. Equation 5.19 is solved for three different types of fluids having the properties and three different configurations of obstacles.

Fluids		Configurations
(a) $n = 0.6$	$K_o = 1.5$ Pa $s^{0.6}$	e = 0 obstacle NE = 288
(b) $n = 1$	$K_o = 0.369$ Pa s	f = 5 obstacle NE = 698
(c) $n = 1$	$K_o = 1.5$ Pa s	g = 9 obstacle NE = 648

Case (a) is a non-Newtonian fluid which follows the Herschel-Bulkley model, while (b) and (c) are Newtonian fluids with different viscosities. The global matrix arising in FEM in the three cases is inverted by method M2. The initial guess of the solution to this matrix inverter is same as the initial condition. The matrices needed for preconditioning are calculated by exact LU decomposition for the first iteration and used in all further iterations.

To see the convergence of PCG for a particular time step a plot of residue as a function of iterations is drawn for the first non-linearity iteration of an arbitrary time step Figure 5.3. In Figure 5.3 the three windows a,b,c shows the three different configurations of the fracture, for fluid (a) namely e, f and g. Figure 5.3 shows that the convergence is monotonic in all the three cases of configuration. Convergence is not affected by the presence of obstacles, but the number of iterations increases. Figure 5.3 also shows that the number of iterations for convergence decreases as the time step increases. This indicates that the present method of supplying guess matrices is quite satisfactory. For the case (b) and case (c) i.e. n=1 there are no non-linearity iterations as the equations are linear.

Figures 5.4, 5.5 and 5.6 shows the flow rate as a function of time for three different configuration e, f and g respectively. Each figure considers three different cases of fluids. Figure 5.4 shows that the time required for reaching steady state in case of fluid (a) is less than fluids (b) and (c) also the flow rate is more than fluids (b) and (c). This is owing to the shear thinning behaviour observed for values of n less than unity. In ease of the Newtonian fluids, an increase in viscosity results in a drop in the flow rate. Figures 5.5 and 5.6 show that as the number of obstacles increases the steady state flow rate decreases in all the three cases of fluids.

Figure 5.7 show the pressure profile for configuration e . The three windows shows fluids (a) , (b) and (c) respectively.The pressure is plotted at different time intervals. Figure 5.7 show that at the earlier time steps the pressure drop is high far a small distance and then the curve is asymptotic. With increasing time steps the pressure curve raises and at steady state it becomes linear. At the same time step the gradient of pressure profile of fluid (c) is more than fluid (b).

Figures 5.8 and 5.9 show the pressure profiles as a

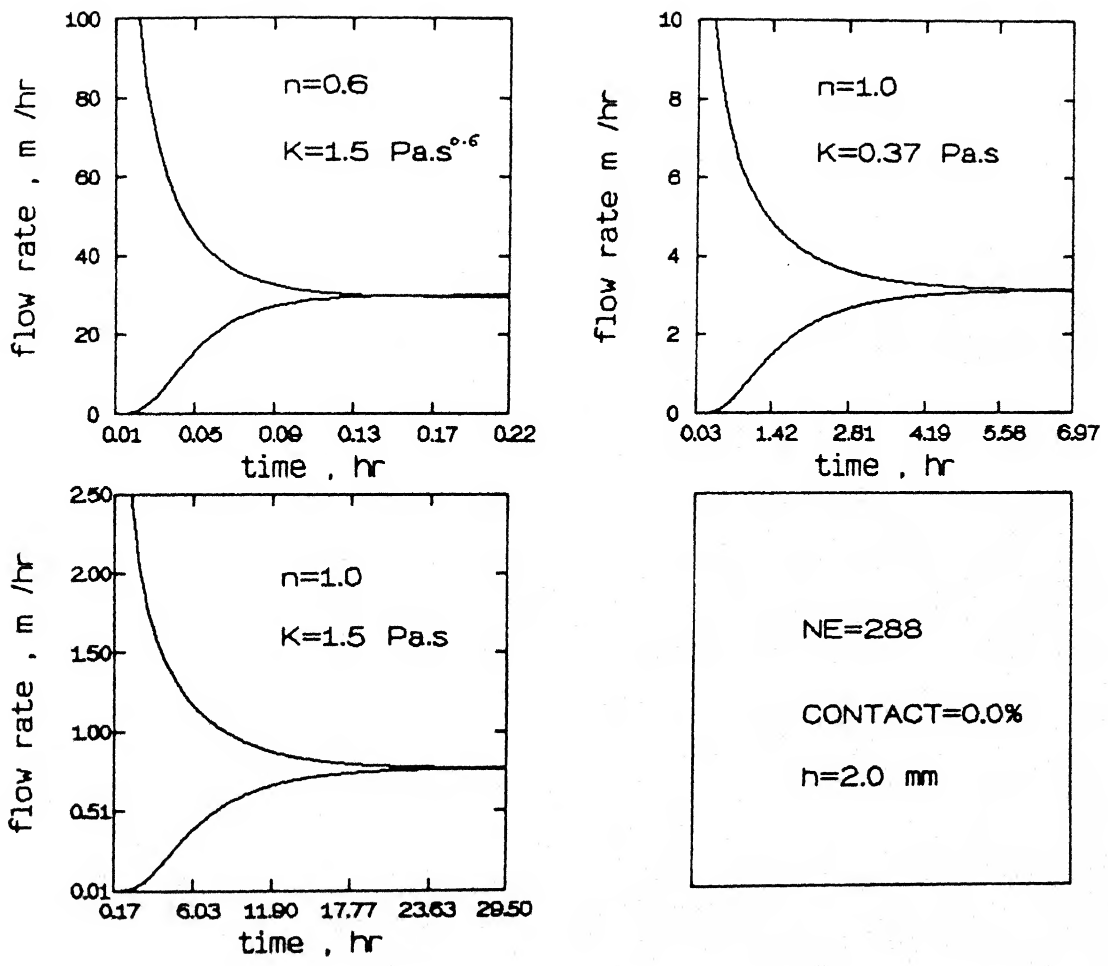


Figure 5.4 : Temporal Variation of Flow Rate at Inflow and Outflow Planes
: Simulation of Oil Flow In Rock Fracture
: No Obstacles

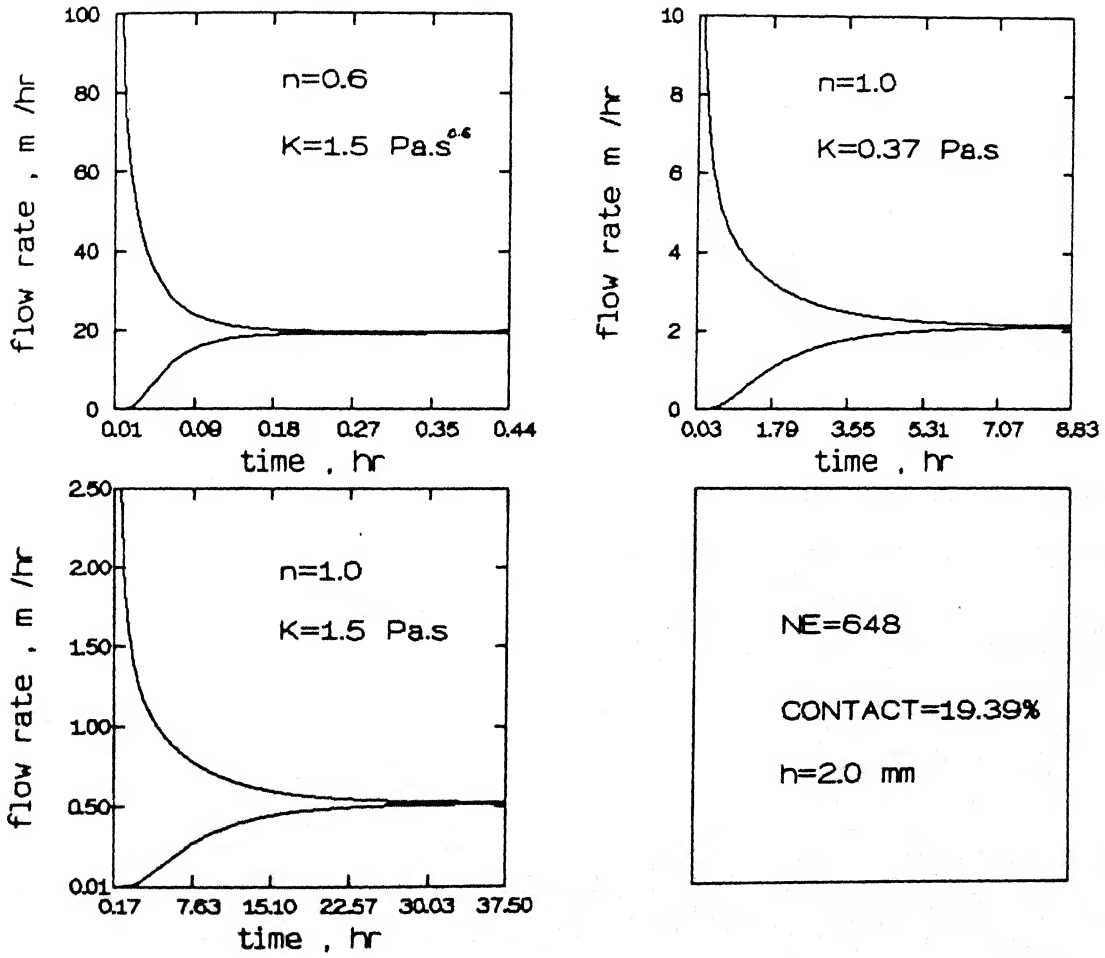


Figure 5.5 : Temporal Variation of Flow Rate at Inflow and Outflow Planes
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: Number of Obstacles 5

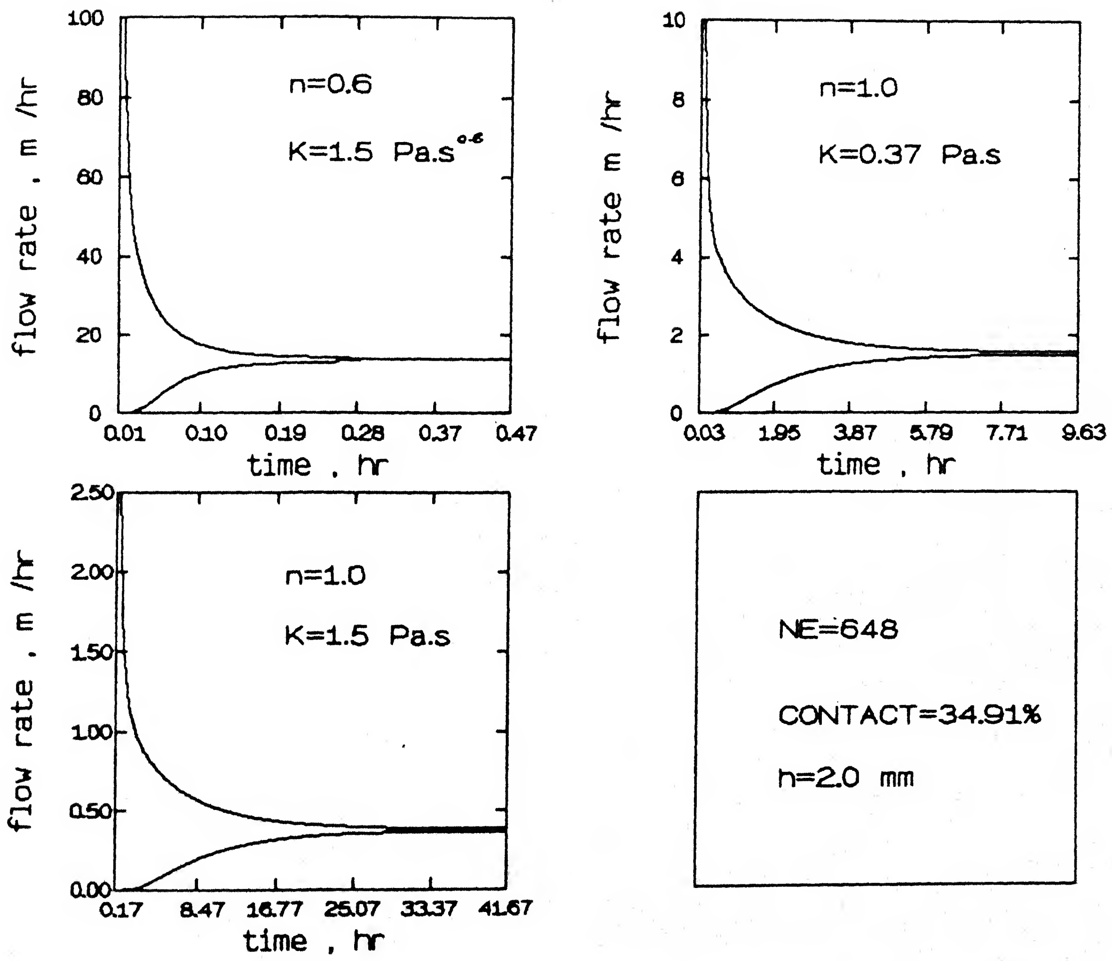


Figure 5.6 : Temporal Variation of Flow Rate at Inflow and Outflow Planes
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: Number of Obstacles 9

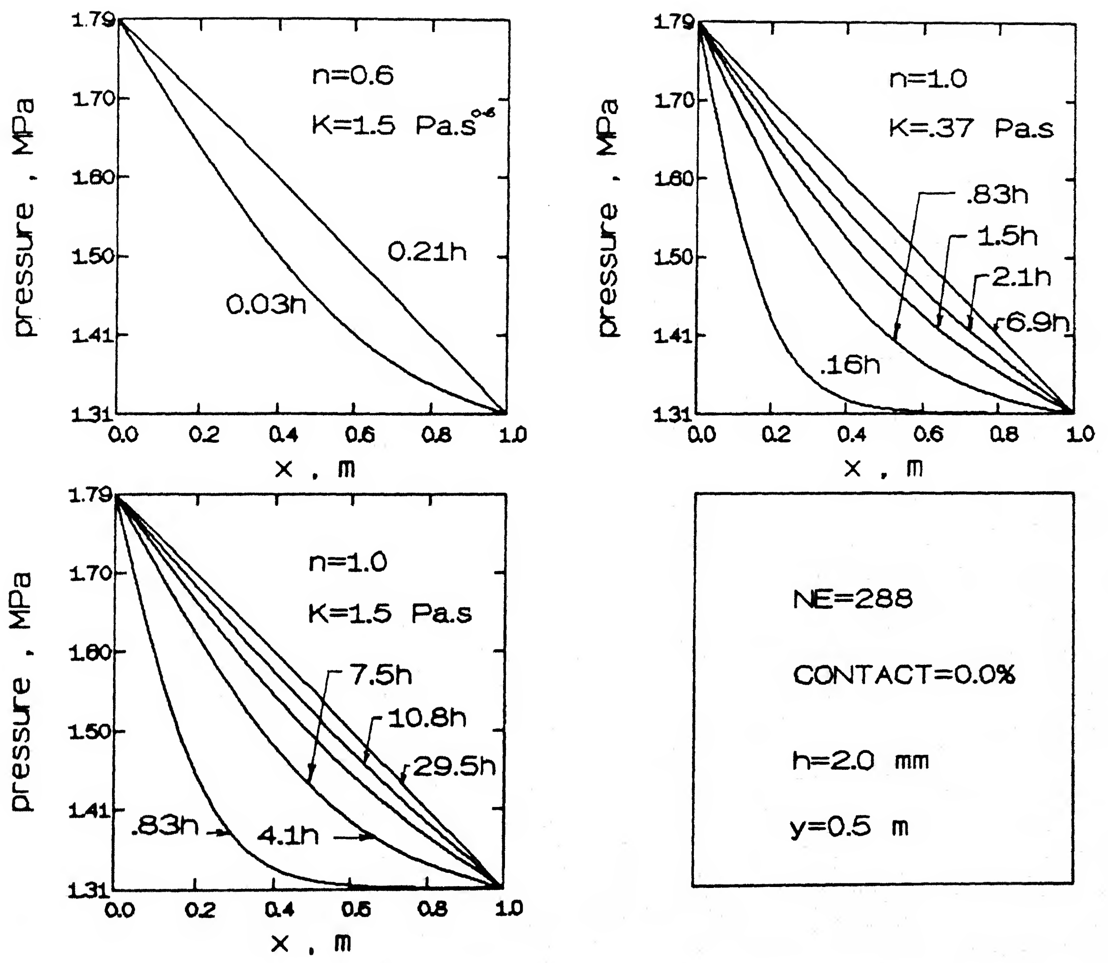


Figure 5.7 : Temporal Variation of Pressure Distribution along Flow Direction
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: No Obstacles

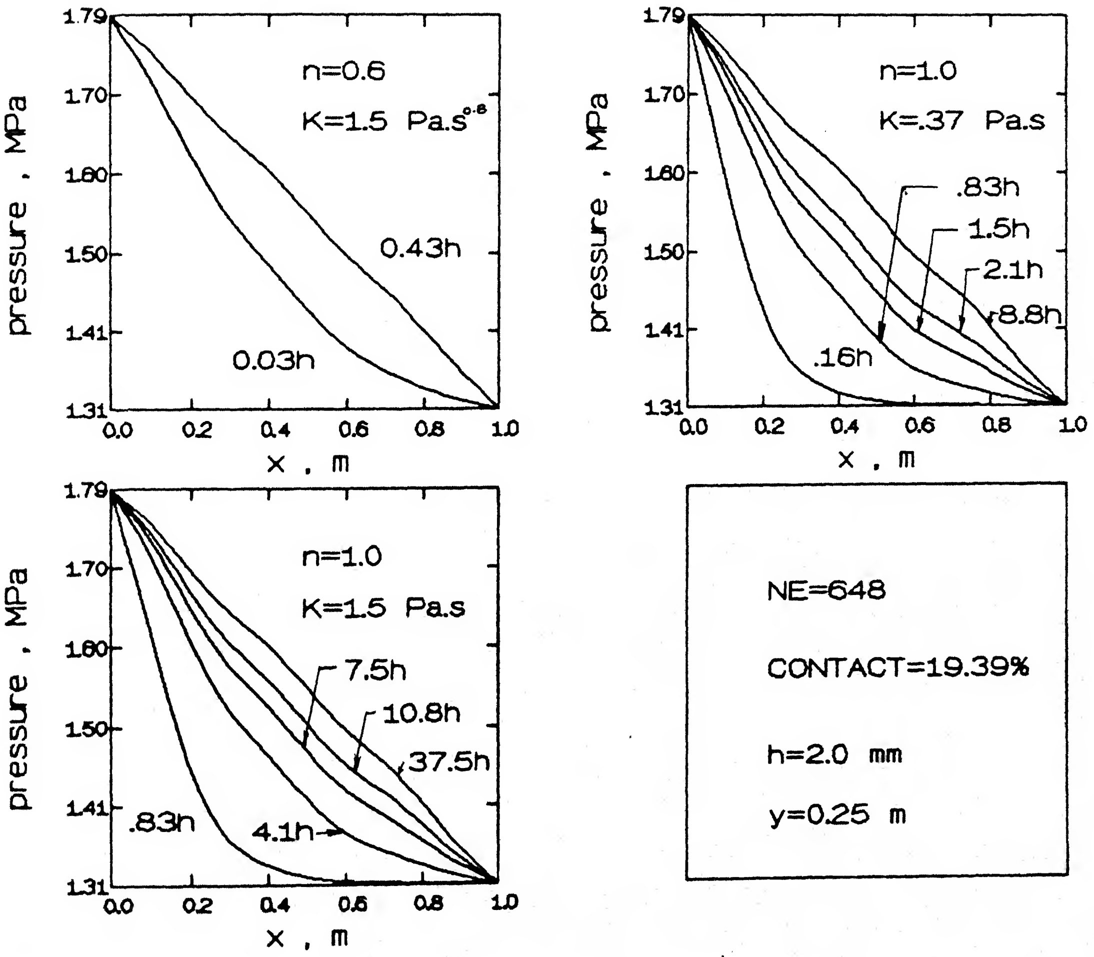


Figure 5.8 : Temporal Variation of Pressure Distribution along Flow Direction
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: Number of Obstacles 5, Along a Plane y = 0.25m

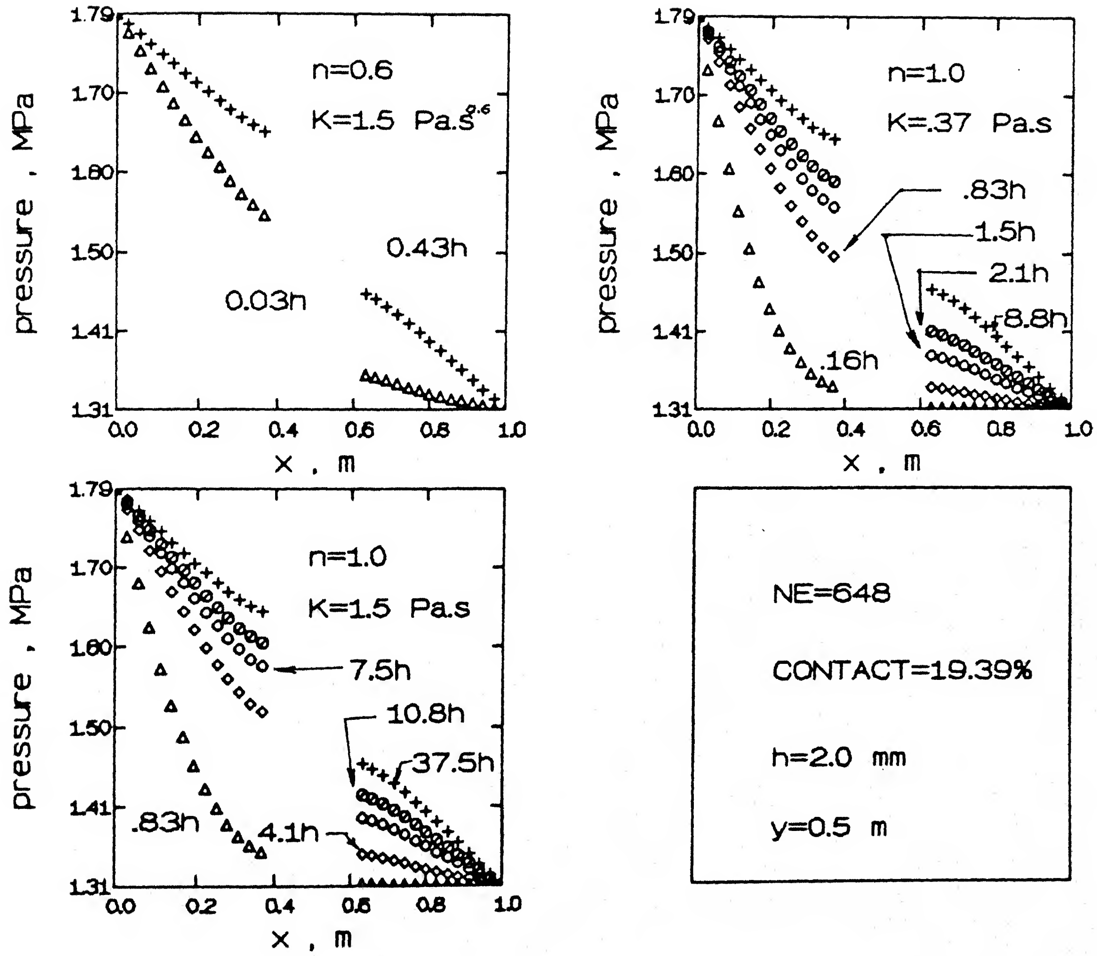


Figure 5.9 : Temporal Variation of Pressure Distribution along Flow Direction
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: Number of Obstacles 5, Along a Plane y = 0.5m

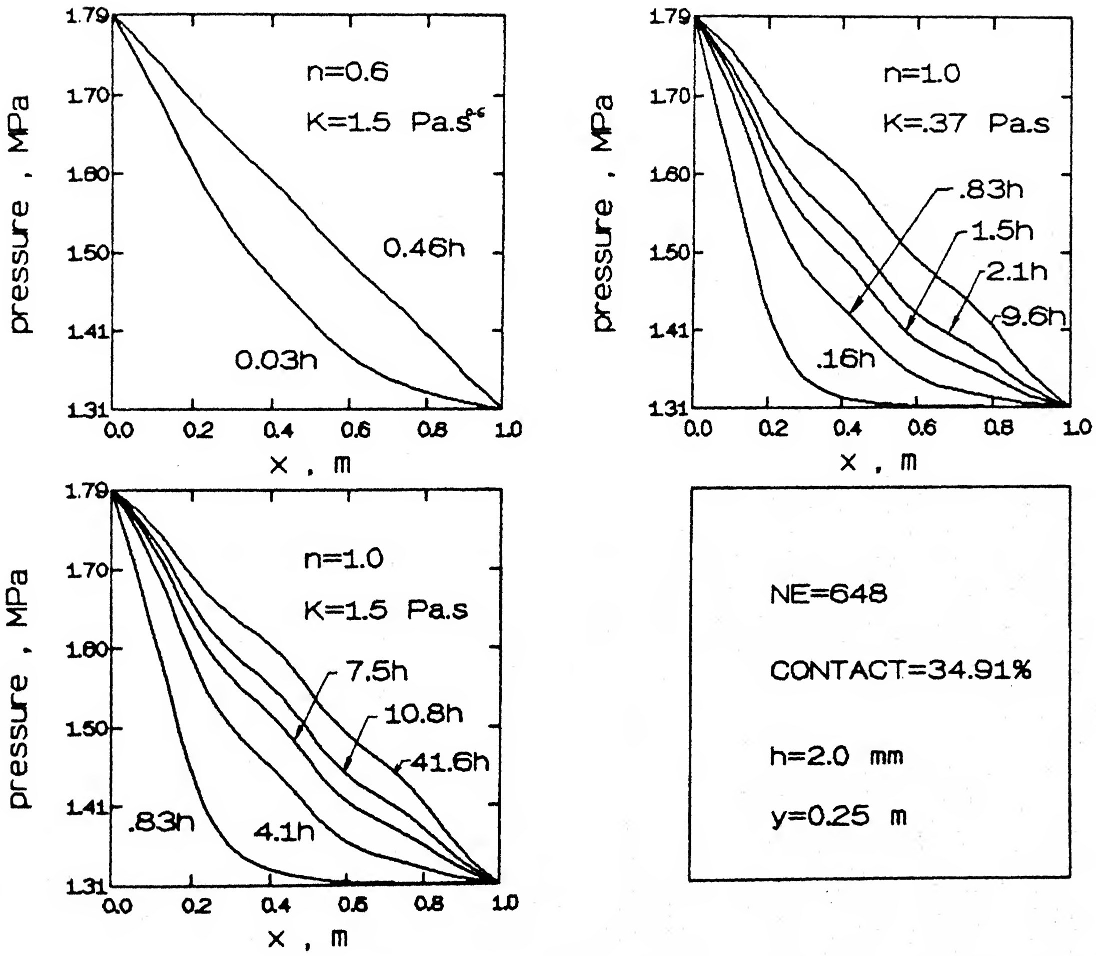


Figure 5.10 : Temporal Variation of Pressure Distribution along Flow Direction
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: Number of Obstacles 9, Along a Plane y = 0.25m

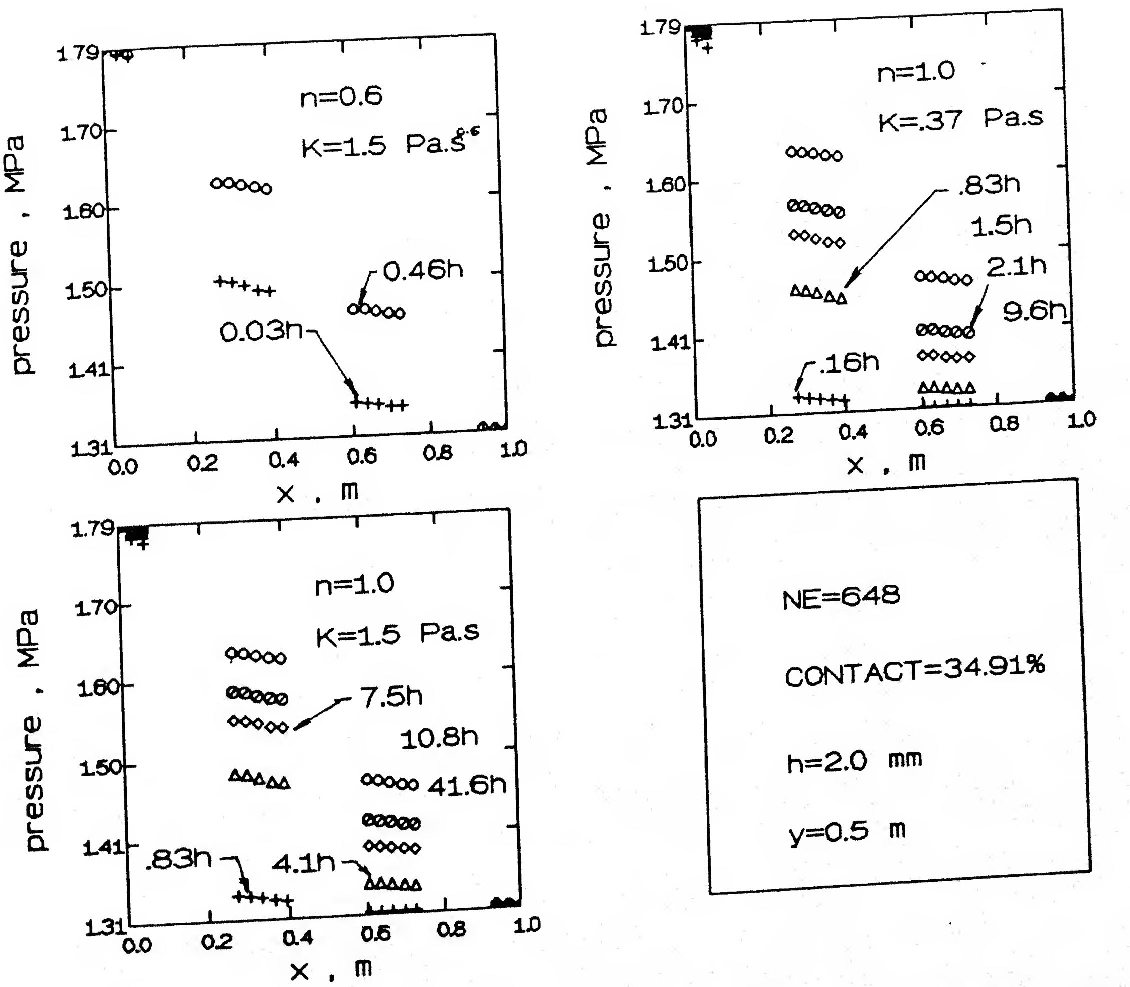


Figure 5.11 : Temporal Variation of Pressure Distribution along Flow Direction
: Simulation of Oil Flow In Rock Fracture Using Newtonian and Non-Newtonian Models
: Number of Obstacles 9, Along a Plane y = 0.5m

function of distance x at y = 0.25 m and y = 0.5 m for configuration f. Figure 5.8 shows the pressure profile at a distance where there is no obstacle.The gradient of the pressure profile is more than the no obstacle case.Figure 5.9 shows the pressure profile where obstacle is present.At the initial time step the pressure profile of Figures 5.8 and 5.9 are same but with increase in time the pressure profile in Figure 5.9,before the obstacle raises slightly above the Figure 5.8 Thus we see that with the presence of the obstacle the pressure near the obstacle increases, but the pressure after the obstacle is not recovered and so the pressure profile in Figure 5.9 is below Figure 5.8.

Figures 5.10 and 5.11 shows the temporal variation of pressure distribution at y = 0.25m and y = 0.5m respectively for the configuration g.The pressure gradient in Figure 5.10 is more than Figure 5.8. This shows that at the same time level the pressure gradient depends on number of obstacle. With increase in number of obstacle the pressure gradient also increases. The Figure 5.11 shows that the pressure distribution after the obstacle is almost linear.

5.5 Flow of Viscoelastic fluid in a Fracture

In this section the flow of viscoelastic fluid with a shear stress, in a fracture is studied. The main stress is on the flow rate at steady state with different value of Yield shear stress.

5.6 Geometric Model and Governing Equation

The geometric model considered here is same as that considered in case of non-Newtonian fluid. The flow behaviour of viscoelastic fluid is given by Herschel-Bulkley model,

$$\tau = K_o \left[\frac{\partial u_o}{\partial z}\right]^n + \tau_o \qquad (5.22)$$

for $\tau > \tau_o$

We know that in case of viscous flow the shear stress is zero at the axis and goes on increasing as we go away from axis towards the wall. Thus around the axis there will be fluid where $\tau = \tau_o$. The stress in this region will be purely elastic and the region will move as solid plug. This type of flow is called *plug flow*.

Figure 5.12 gives the velocity profile of a viscoelastic fluid. From a force balance one obtains,

$$2.\tau = 2 \cdot \frac{\partial p_o}{\partial x} . z \qquad (5.23)$$

By substituting equation (5.22) into equation (5.23) and solving for velocity using the boundary conditions,

$$z = h, \quad u = 0 \qquad (5.24)$$

one obtains

$$u_o = \left[\frac{n}{n+1}\right]\left[\frac{\partial p_o}{\partial x} \cdot \frac{1}{K_o}\right]^{\frac{1}{n}} \left\{ \left[z - \tau_o / (\partial p_o / \partial x)\right]^{\frac{(n+1)}{n}} + \left[h - \tau_o / (\partial p_o / \partial x)\right]^{\frac{(n+1)}{n}} \right\}$$

(5.25)

$$v_o = \left[\frac{n}{n+1}\right]\left[\frac{\partial p_o}{\partial y} \cdot \frac{1}{K_o}\right]^{\frac{1}{n}} \left\{ \left[z - \tau_o / (\partial p_o / \partial y)\right]^{\frac{(n+1)}{n}} + \left[h - \tau_o / (\partial p_o / \partial y)\right]^{\frac{(n+1)}{n}} \right\}$$

(5.26)

To eliminate the z-coordinate these equations (5.25) and (5.26)

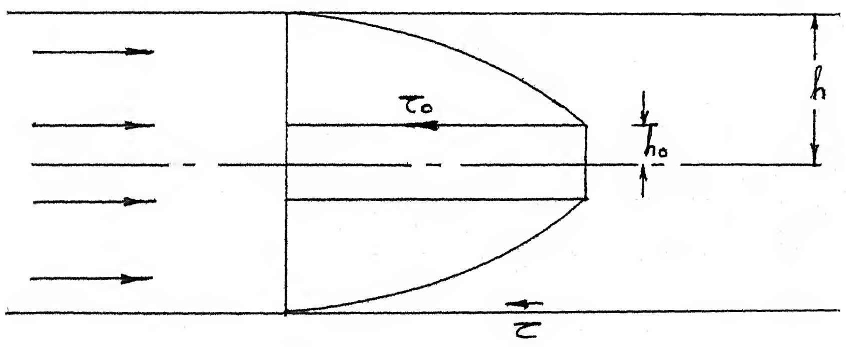


Figure 5.12 : Velocity Profile for the Channel Flow of
: A Non-Newtonian Fluid with Yield Stress.

are depth averaged over the fracture aperture. Then these average velocities are

$$\bar{u}_o = \alpha \left[\frac{\partial p_o}{\partial x}\right]^{\frac{1}{n}} \left[h - \tau_o/(\partial p_o/\partial x)\right]^{\frac{(2n+1)}{n}} \qquad (5.27)$$

$$\bar{v}_o = \alpha \left[\frac{\partial p_o}{\partial y}\right]^{\frac{1}{n}} \left[h - \tau_o/(\partial p_o/\partial y)\right]^{\frac{(2n+1)}{n}} \qquad (5.28)$$

$$\therefore \quad \alpha = \left[\frac{n}{2n+1}\right] \left[\frac{1}{K_o}\right]^{\frac{1}{n}} \cdot \frac{1}{h} \qquad (5.29)$$

$$\frac{\partial \bar{u}_o}{\partial x} + \frac{\partial \bar{v}_o}{\partial y} = \xi_o \cdot \frac{\partial p_o}{\partial t} \qquad (5.30)$$

Substituting equations (5.27) and (5.28) into equation (5.30) we get the final form of pressure equation as,

$$\frac{\partial}{\partial x}\left\{\left[\frac{\partial p_o}{\partial x}\right]^{\frac{1}{n}} \cdot \left[h-\tau_o/(\partial p_o/\partial x)\right]^{\frac{2n+1}{n}}\right\} + \frac{\partial}{\partial y}\left\{\left[\frac{\partial p_o}{\partial y}\right]^{\frac{1}{n}} \cdot \left[h-\tau_o/(\partial p_o/\partial y)\right]^{\frac{2n+1}{n}}\right\}$$

$$= \frac{\xi_o}{\alpha} \cdot \frac{\partial p_o}{\partial t} \qquad (5.31)$$

The boundary conditions for p_o are taken as,

$x = 0, \quad p_o = 1.793$ MPa

$x = L, \quad p_o = 1.31$ MPa.

On the obstacles, $\frac{\partial p_o}{\partial x} = \frac{\partial p_o}{\partial y} = 0$ as $u_o = v_o = 0$ (5.32)

initial condition is

$t = 0 \qquad p_o = 1.31$ MPa

However, as before, to avoid numerical oscillators, a mild

Table 5.2 Properties of a Power-Law Fluid with Yield Stress

Compressibility of oil (ζ_o)	0.03447 Pa^{-1}
Power-law Index (n)	0.6
Yield Stress (τ_o)	3.5 Pa
Consistency Index (K_o)	1.5 $Pa.s^{0.6}$
Inlet Pressure (P_1)	1.793 MPa
Exit Pressure (P_2)	1.31 Mpa

pressure gradient is applied at t = 0.

The values of the various constants in the Equation (5.31) are given in Table 5.2. The **FEM** formulation of Equation (5.31) is given in Appendix-c.

5.7 Results

Equation (5.31) is solved till steady state is reached, for various values of the yield shear stress. Figure 5.13 shows the plot of flow rate as a function of shear stress. At zero value of shear stress the flow rate is same as the non-Newtonian fluid shown in Figure 5.4. As the value of shear stress increases there is reduction in the flow rate. This is due to more resistance to flow. With increase in the value of shear stress the plug flow increases but the average velocity of flow decreases. This inturn decreases the flow rate. Figure 5.12 shows that there is linear decrease in the flow rate with increase in yield shear stress.

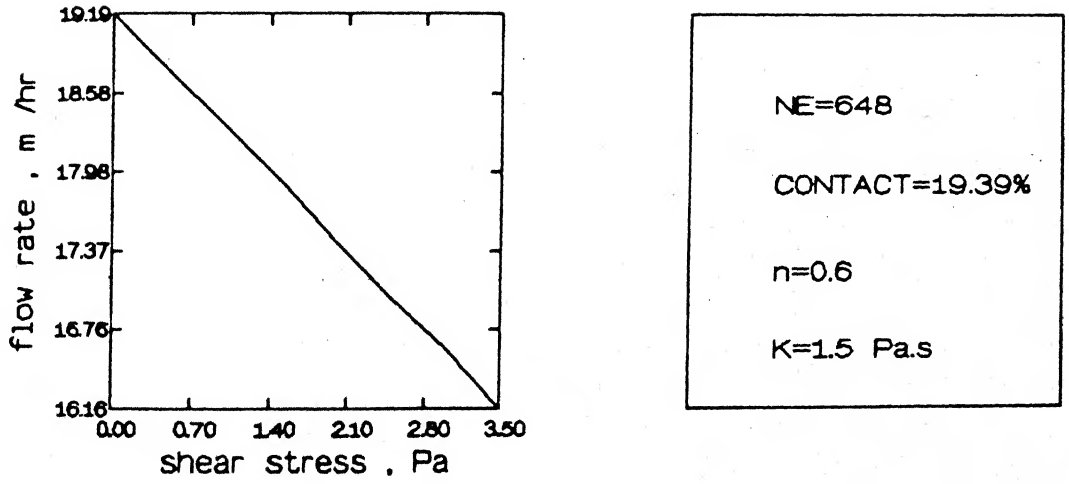


Figure 5.13 : Variation of Flow Rate with Yield Shear Stress
: Simulation of Oil Flow In Rock Fracture Using Non-Newtonian Model
: Number of Obstacles 5

CHAPTER 6

SIMULATION OF OIL RECOVERY USING WATER INJECTION METHOD

6.1 Introduction

The efficiency of a matrix inversion algorithm depends upon its capacity to handle an ill-conditioned asymmetric matrix. One generally comes across ill-conditioned asymmetric matrices while solving fluid flow problems. A problem of intermediate level of complexity is flow of two liquids, oil and water in a rock fracture. In the present chapter oil recovery using high pressure water injection has been numerically studied. The matrices arising from the system of governing differential equations has been inverted using Preconditioned Conjugate Gradient method (M2).

Oil is an important source of energy for sustenance of modern society. It is clear that there will be a severe oil shortage in the near future. Hence recovery of oil from existing reservoirs has gained importance. It is known that a large portion of oil still remains unrecovered. This is because until recently, there was no real economic incentive to develop enhanced oil recovery (EOR) technologies. Of late, much research has been directed towards developing EOR techniques. In the present chapter, the numerical simulation of an EOR technique is presented.

Before the advent of enhanced oil recovery techniques, conventional methods (primary and secondary) were used for oil production.

(a) **Primary Recovery**: - In this case the presence of oil in the reservoir is high and no pumping is required to force the oil. 15% - 20% extraction of oil is possible in this way.

(b) **Secondary Recovery**: - In this case the oil pressure in the reservoir is maintained by injecting water, air or steam and the oil is physically displaced towards the well. 50% extraction of oil is done in this manner.

In order to recover more oil several enhanced recovery techniques involving complex chemical and thermal effects have been developed. Oil is trapped in the fracture due to surface tension, several EOR techniques involves lowering of surface *tension,* e.g. thermal recovery methods (*Boberg* 1988) reduces oil viscosity by raising its temperature and thus displacement efficiency of oil is improved. Polymer flooding (*Boberg*, 1988 *andShah*, 1977) is used to increase the volume of the reservoir contacted by the displacing fluid. Surfactant flooding reduces the inter facial tension between oil and water considerably (*Muralidhar and Chatterjee*, 1994). When oil is trapped in rocks it is common to induce fractures in it and subsequently extract oil by high pressure water flooding. Simulation presented in this chapter pertains to this branch of oil recovery.

Reservoir simulation is one of the very challenging problems in engineering (*Ewing*, 1983)and (*Douglas*,1983). The governing partial differential equations are highly nonlinear and coupled due to complexity of the geometry, complex nature of the constitutive relationships and the instability of the oil-water interface. While analytical solutions are difficult to obtain, general purpose numerical methods exhibit instability and consequent divergence. It is the topic of extensive current research.

6.2 Definitions.

Percentage pore volume (PPV)

It is an index of efficiency of oil recovery. It is the ratio of total volume of oil produced to the total pore volume.

saturation.

It is the fraction of the space available to flow occupied by a phase

6.3 Geometric and Mathematical Model.

The geometric model consider in this chapter is similar to that in Chapter 5. A rock fracture is modeled as the gap betweenparallel plates with a distribution of contact areas. The equations are depth-averaged and hence two dimensional in the fracture plane. The rectangular domain of the fracture is taken to have a nominal size of 1m x 1m x 2mm, the smallest dimension being the aperture of the fracture. In a cartesian plane $x = o$ and $x = L$ are the inflow and outflow planes respectively. The side boundaries of the domain namely $y = o$ and $y = L$ are symmetry planes and hence do not permit a net normal velocity, as shown in Figure 6.1.

The dynamics of two component flow through the fracture media is too complicated to be described at microscopic scale. Hence equations applicable at a macroscopic scale have been used in the

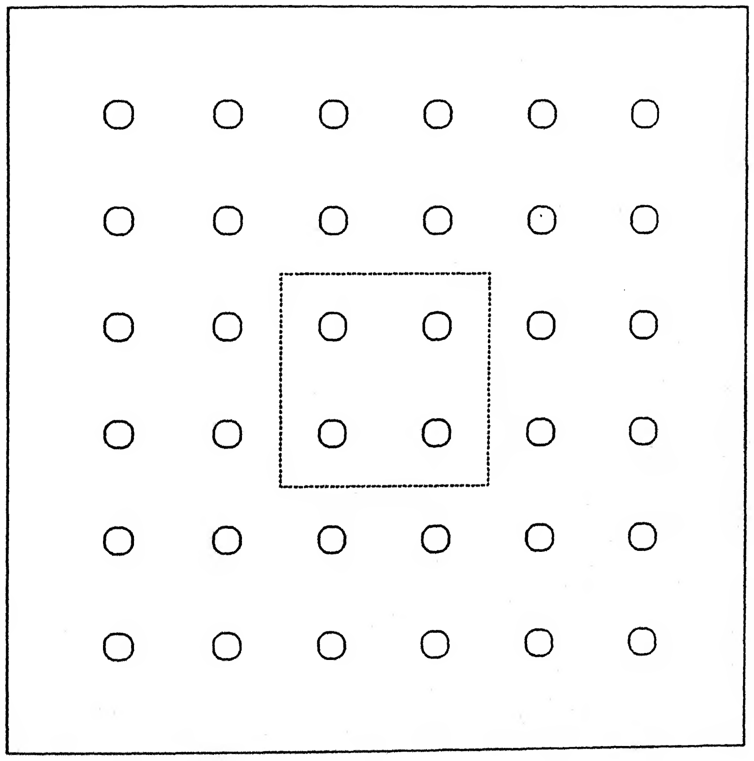


Figure 6.1 : Distribution of Contact Areas in A Large Fracture present study. A formulation involving oil and water pressures has been adopted in this work. The governing equations are derived in Appendix-D. In the present study oil is model led as a power law fluid while water is taken to be Newtonian. The pressure equations for oil and water are given below:

Oil;

$$\frac{\partial p_o}{\partial t} \cdot \left[S_o \zeta_o - \frac{dS_w}{dp_c} \right] + \frac{dS_w}{dp_c} \cdot \frac{\partial p_w}{\partial t} = \left[\frac{n}{2n+1}\right]\left[\frac{1}{K_o}\right]^{1/n} \cdot \left[h^{(n+1)/n}\right]$$

$$\frac{\partial}{\partial x}\left[K_{ro}\left[\frac{\partial p_o}{\partial x}\right]^{1/n}\right] + \frac{\partial}{\partial y}\left[K_{ro}\left[\frac{\partial p_o}{\partial y}\right]^{1/n}\right] \qquad (6.1)$$

$$C_1 \cdot \frac{\partial p_o}{\partial t} + C_2 = C_3 \left[\frac{\partial}{\partial x}\left[\frac{\partial p_o}{\partial x}\right]^{1/n} + \frac{\partial}{\partial y}\left[\frac{\partial p_o}{\partial y}\right]^{1/n}\right] \qquad (6.2)$$

where

$$C_1 = S_o \, \xi_o - \frac{dS_w}{dp_c} \qquad (6.3)$$

$$C_2 = \frac{dS_w}{dp_c} \cdot \frac{\partial p_w}{\partial t} \qquad (6.4)$$

$$C_3 = \left[\frac{n}{2n+1}\right]\left[\frac{1}{K_o}\right]^{1/n}\left[h^{(n+1)/n}\right] . \; K_{ro} \qquad (6.5)$$

Water;

$$\frac{\partial p_w}{\partial t} \cdot \left[S_w \zeta_w - \frac{dS_w}{dp_c} \right] + \frac{dS_w}{dp_c} \cdot \frac{\partial p_o}{\partial t}$$

$$= \left[\frac{h^2}{3.K_w}\right]\left\{\frac{\partial}{\partial x}\left[K_{ro}\frac{\partial p_o}{\partial x}\right] + \frac{\partial}{\partial y}\left[K_{ro}\frac{\partial p_o}{\partial y}\right]\right\} \qquad (6.6)$$

$$K_1 \cdot \frac{\partial p_w}{\partial t} + K_2 = K_3 \left[\frac{\partial^2 p_W}{\partial x^2} + \frac{\partial^2 p_W}{\partial x^2}\right] \qquad (6.2)$$

where

$$K_1 = S_w \xi_w - \frac{dS_w}{dp_c} \qquad (6.3)$$

$$K_2 = \frac{ds_w}{dp_c} \cdot \frac{\partial p_o}{\partial t} \qquad (6.4)$$

$$K_3 = \frac{h^2}{3.K_w} \cdot K_{rw} \qquad (6.5)$$

Equations (6.2) and (6.7) are strongly coupled nonlinear partial differential equations. Their form is that of a parabolic heat conduction equation with a source term namely,

$$\frac{dS_w}{dp_c} \left[\frac{\partial p_o}{\partial t} - \frac{\partial p_w}{\partial t}\right]$$

The constitutive relations needed to solve equation (6.2) and (6.7) are the following:

Capillary pressure : $p_c = p_o - p_w = p_c\ (S_w)$

Relative permeability

Oil : $K_{ro} = K_{ro}\ (S_w)$

Water : $K_{rw} = K_{rw}\ (S_w)$

The following functional form of p_o, K_{ro}, K_{rw} have been used in the present work.

$$p_c = \frac{1-S_w}{S_w} \text{, where } p_c \text{ is in KPa} \qquad (6.11)$$

$$K_{ro} = 1 - \exp\left[-\frac{1}{S_w}\right] \qquad (6.12)$$

$$K_{rw} = 0.3\ S_w \qquad (6.13)$$

These correctly reproduce the constraint

$p_c \longrightarrow \infty$ as $S_w \longrightarrow 0$, $K_{rw} \longrightarrow 0$, $K_{ro} \longrightarrow 1$

p_c —— 0 as $S_w \longrightarrow 1$, $K_{rw} \longrightarrow 0.3$, $K_{ro} \longrightarrow 0.632$.

6.4 Initial and Boundary Conditions

At the left hand boundary of the fracture domain water at a pressure of 1.793 MPa is applied. Instantaneously the saturation of water at this plane increases to 0.9. This value is not unity because of some residual oilwhich because of chemical bonding cannot be flushed out. It is the maximum water saturation possible anywhere in the domain. At the right boundary oil pressure is 1.31 MPa, saturation of water is 0.1, maximum oil saturation is 0.9. Water pressure is 1.301 MPa. Initially the fracture region is maintained at a uniform oil pressure of 1.31 MPa and the oil saturation is 0.9. Water pressure is obtained by combining oil pressure and capillary pressure.

The initial condition for p_o, p_w and S_w have been smoothened to avoid the computational oscillations resulting from a step like initial condition. The modified conditions are given below:

At time t = 0

$$S_w\,(x,y) = 0.9 - 8x \qquad 0 \le x \le 0.1\,L \qquad (6.14)$$

$$p_w\,(x,y) = 1.793 - 4.92\,x \qquad 0 \le x \le 0.1\,L$$

$$= 1.301 \text{ MPa} \qquad 0.1 < x \le L$$

At t > 0, the impermeability conditions at the bounding surfaces of the host rock give,

$u_o(x,y) = v_w\,(x,y) = 0$ at $y = 0, L$

These conditions in terms of pressure can be found by equation

(D.2) and (D.5), as

$$\frac{\partial p_o}{\partial y} = \frac{\partial p_w}{\partial y} = 0 \quad \text{at } y = 0,\ L \qquad (6.15)$$

The boundary conditions at x=0 and x=L are summarized below.

$x = 0$

$p_w = 1.793$ MPa

$S_w = 0.9$

$p_o = p_w + p_c = 1.793 + 0.00011 = 1.79311$ MPa

$x = L$

$p_o = 1.31$ MPa

$S_o = 0.9$

$p_w = p_o - p_c = 1.31 - 0.009 = 1.301$ MPa

6.5 Solution of Pressure Equations

Equations 6.2 and 6.7 and associated boundary conditions (6.16) have been solved by Galerkin FEM. A bounded matrix is obtained for oil and water equations separately. These matrices are inverted by Preconditioned Conjugate Gradient method (M2) described in Chapter 4. The guess matrices L and U are calculated only during the first time step and then they are stored and are used for further computation. The initial guess value of solution to the PCG is same as the initial value of p_o and p_w. This way of supplying L and U shows a good convergence.

The system of equations governing the distribution of oil and water pressure are non-linear and mutually coupled. They are solved simultaneously and by iteration. The algorithm for solving oil and water pressures is as follows:

1. The values of p_o, p_w and S_w are given at t = o.

2. The coefficients C1, C2, C3 and K1, K2, K3 of the pressure equations 6.2 and 6.7 respectively are computed using the values of p_o, p_w and S_w at the last available time level.

3. The oil equation is solved to obtain the approximate value of p_o at the new time level.

4. The new value of water is used to solve pressure equation of oil.

5. Steps 3 - 5 are repeated till convergence in oil and water pressures at the latest time level.

6. Values of coefficients C1, C2, C3 and K1, K2, K3 are recalculated for next time level and steps 3 - 6 are repeated.

6.6 Results

The oil and water pressure equations have been solved for two configurations. In the first case there are no obstacles in the fracture. For this case a coarse grid (NE = 288) is used. The movement of the water saturation front is studied with respect to time and PPV is plotted as a function of time. In the second case nine obstacles are placed as shown in Figure 5.2. The total contact area is 34.9%. A finer grid (NE = 648) has been used for this problem. The effect of obstacles on the oil removal rate is studied. The oil and water properties used in the numerical simulations are given in Table 6.1. Figure 6.2 shows the water saturation profile at different times. This figure shows that the water front moves fast and drives the oil out of the fracture.

Table 6.1 Properties of a Oil and Water

Compressibility of Oil (ζ_o)	0.03447 Pa^{-1}
Compressibility of Water (ζ_w)	0.02137 Pa^{-1}
Power-law Index (n)	0.6
Viscosity of Water at 30°C (K_w)	79.77e-5 $Pa.s$
Consistency Index in Herschel-Bulkley Model (K_o)	1.5 $Pa.s^{0.6}$
Injection Water Pressure (P_1)	1.793 MPa
In-situ Oil Pressure (P_2)	1.31 Mpa

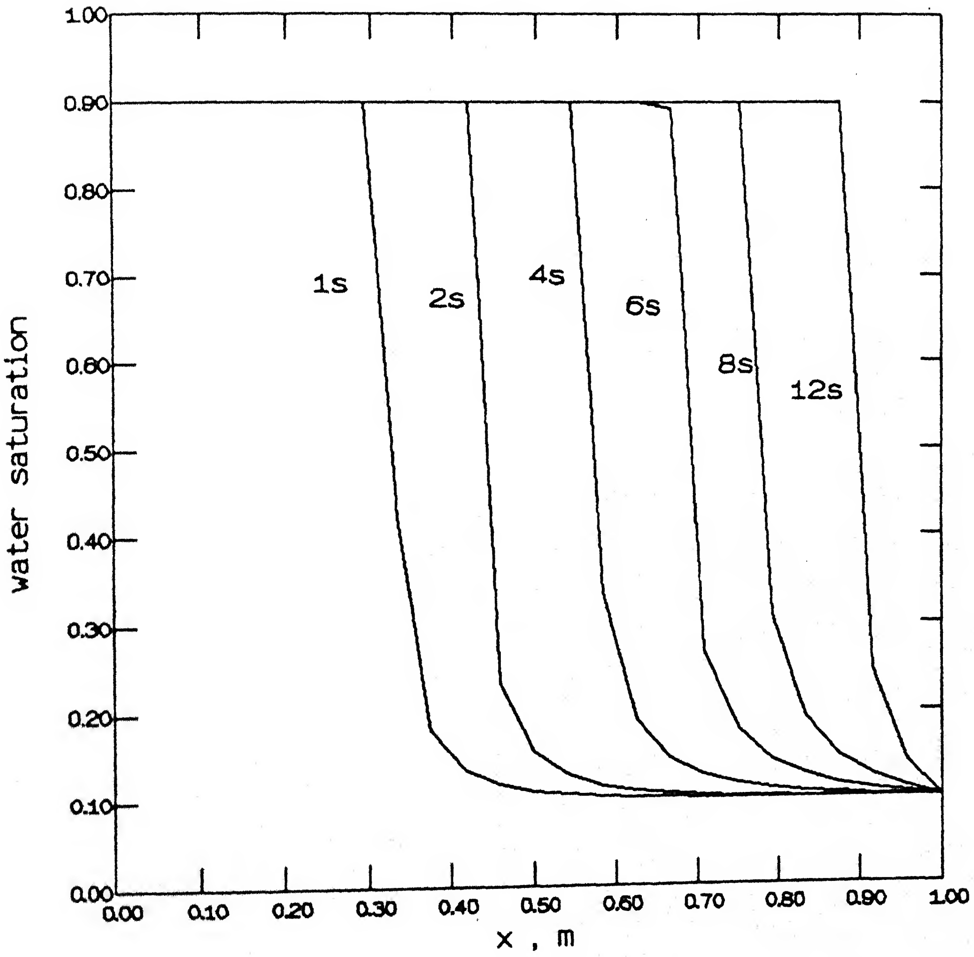


Figure 6.2 : Propagation of Water Saturation Front with Time
: Simulation of Oil Recovery Using Water Injection Method
: No Obstacles

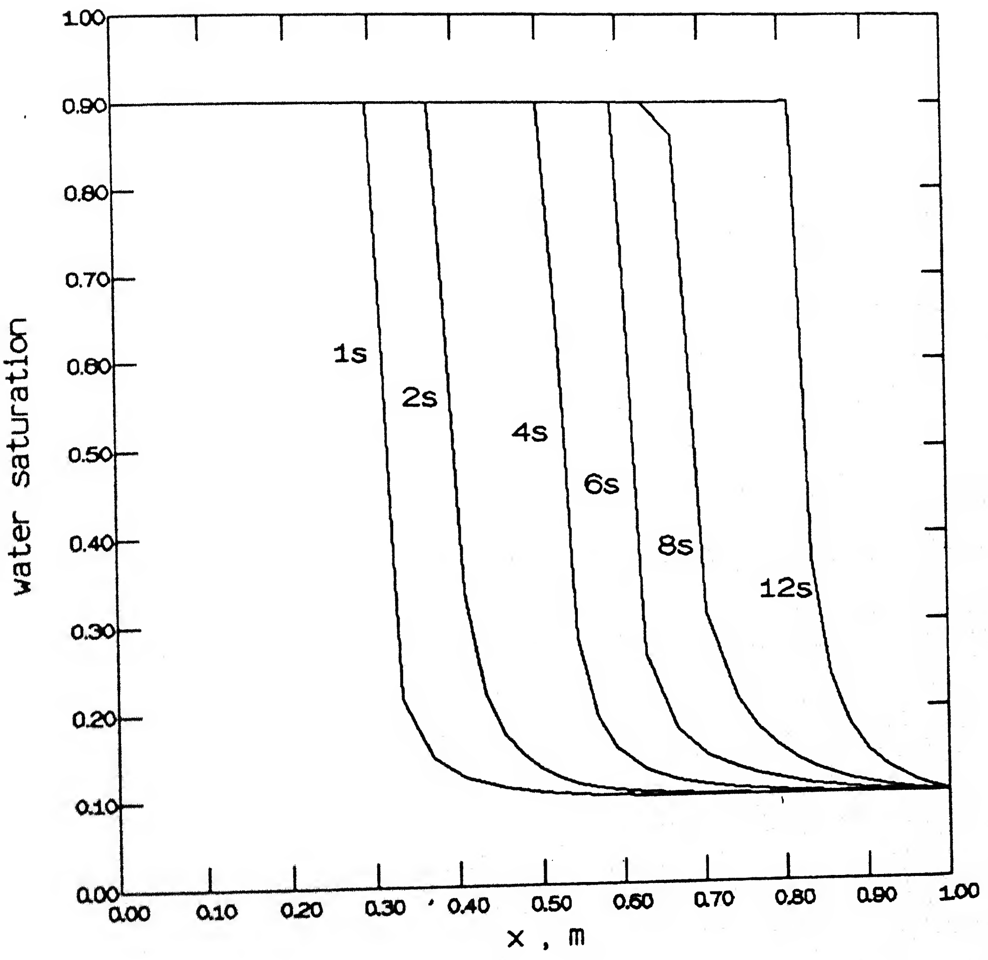


Figure 6.3 : Propagation of Water Saturation Front with Time
: Simulation of Oil Recovery Using Water Injection Method
: Number of Obstacles 9

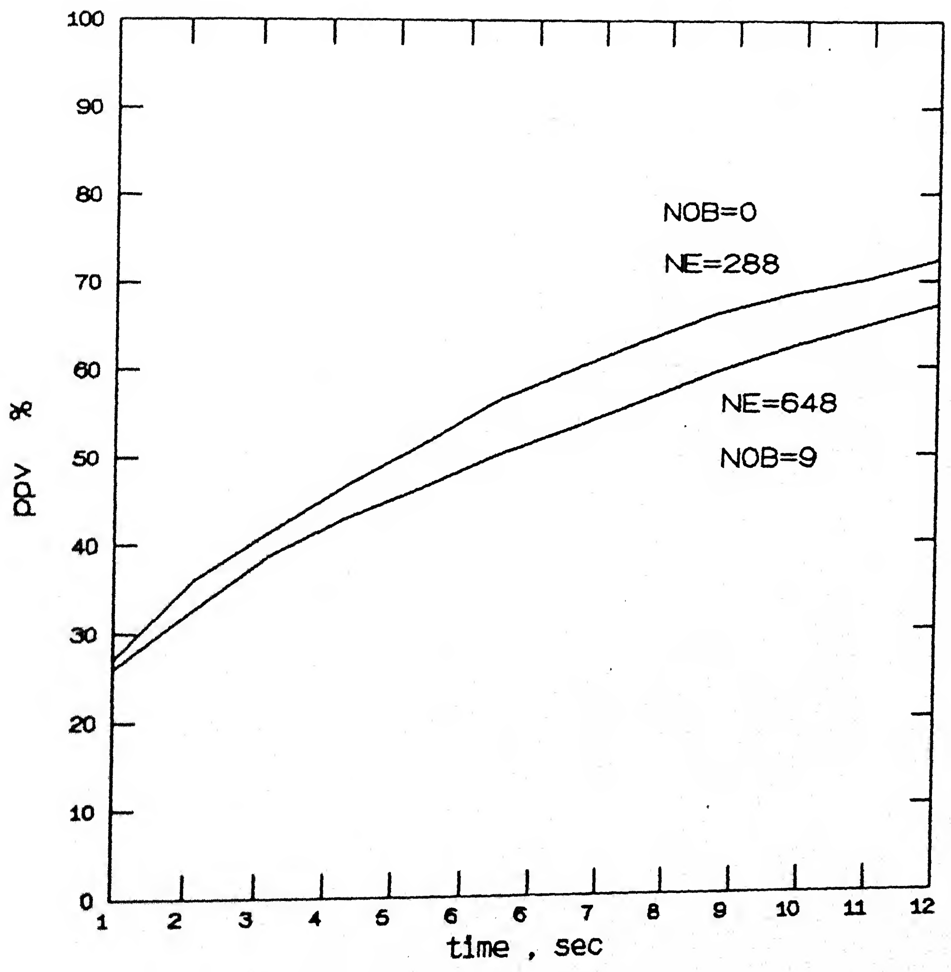


Figure 6.4 : Temporal Variation of Percentage Pore Volume
: Simulation of Oil Recovery Using Water Injection Method

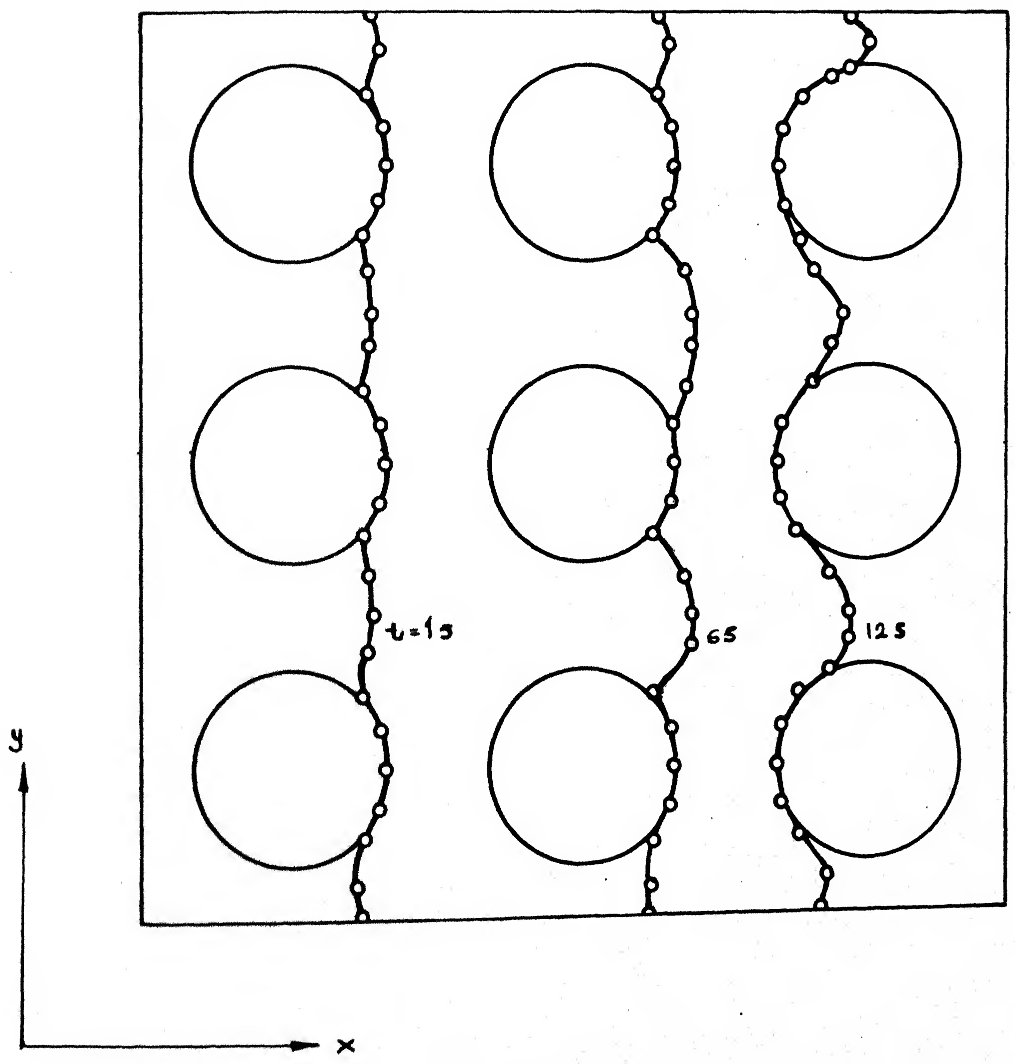


Figure 6.5 : Movement of Water Saturation Front At Different Time Intervals

Figure 6.3 also shows the water saturation profile when obstacles are present in the fracture. By comparing Figure 6.2 and Figure 6.3 we see that the front moves slowly. This is due to additional resistance caused by obstacles. Figure 6.4 shows a plot of PPV versus time for a fracture with and without contact areas. This figure shows that the PPV decreases as the number of obstacles in the fracture increases. Figure 6.5 shows the water saturation front when nine obstacles are present.

CHAPTER 7

CONCLUSIONS AND SCOPE FOR FUTURE WORK

A variety of steady and transient diffusion dominated problems have been considered in the present study. The matrix equation arising from a finite element formulation has been inverted using a Preconditioned Conjugate Gradient Method. Results show that a preconditioner based on LU decomposition of the unsymmetric matrix generated for small time (Method M2) is satisfactory for all future time.

Future work should concentrate on the following topics.

1. Testing of Method M2 for Navier-Stokes problem.
2. Parallelization of M2 on commercial parallel machines.
3. Application of M2 for matrices generated by alternative FEM formulation.

REFERENCES

Amodio, P and Brugnano, L *A Parallel Version of Some Block Preconditionings.* Impact of Computing in Science and Engineering, 3, pp 235-243, 1991.

Al-Fariss, T. and Pinder, K.L. *Flow Through Porous Media of a Shear Thinning Liquid with Yield Stress.* The canadian journal of Chemical Engineering, vol 65, pp 391-405, 1987.

Axelsson, O *A Class of Iterative Methods of Finite Element Equations,* Computer Methods in Appl. Mech. and Engg. 9, pp 123-137, 1976.

Boberg, T.C, *Thermal Methods of Oil Recovery,* Exxon Monograph, John Wiley, 1988.

Brugnano, L and Trigiante, D *A Full Preconditioner for a class of M-matrices.* Appl. Math. Lett., vol 4, No. 1, pp13-15, 1991.

Chatterjee, A and Muralidhar, K *Numerical Study of Enhanced Oil Recovery using Surfactants,* to appear in Int. J. of Num. Methods in Heat and Fluid Flow, 1995.

Crochet, M.J. Davies, A.R. Walters, K *Numerical Simulation of Non-Newtonian Flow* .Elsevier 1984.

Duff, I.S. *MA28 - A Set of Fortran Subroutines for Sparse Unsymmetric Linear Equations.* AERE, Harwell, U.K, 1980.

Eving, R *Mathematics of Reservoir Simulation,* SIAM Publications on Appl. Math., 1983.

Fox. L, Huskey, D and Wilkinson, J, *Notes on the Solution of Algebraic Linear Simultaneous Equations,* Quart.J. of Mech. and Appl. Math. 1, pp 149-173, 1948.

George, A. and Kunio, H. *The Precondition Conjugate Gradient Method on the Hypercube.* ACM, pp 1676 - 1686. 1988.

Golub, G.H. and O'leary, D. P. *Some History of the Conjugate Gradient and Lanczos Algorithms 1948 - 1976.* SIAM Review,vol. 31, No. 1, pp 50 - 102. 1989.

Hackbusch, W *A parallel Conjugate Gradient Method,* J.of Numerical Linear Algebra with Applications, vol 1, No. 2, pp122-147, 1992.

Hestenes, Magnus, R, and Stiefel, E *Methods of Conjugate Gradients for Solving Linear Systems,* J. Res. Nat. Bur. Standards 49, pp 409-436, 1952.

Jim Douglas Jr., *Finite Difference Methods for Two Phase Incompressible Flow in Porous Media,* SIAM J. Num. Anal., vol 20, 1983.

Kershaw, D.S. *The Incomplete Cholesky Conjugate Gradient Method for the Iterative Solution of Systems of Linear Equations.* J.of.comp.physics 26, pp 43-65,1978.

Kronsjo, L.I. *Algorithms:Their Complexity and Efficiency.*Jhon Wiley and Sons.1979.

Muralidhar, K. and Pillai, K.M. *A Numerical Study of Oil Recovery Using Water Injection Method.*Num. Heat transfer.(A) 24, 305-322 ,1993

Ortega, J.M. *Intoduction to Parallel and Vector Solutions of Linear Systems.* Plenum,New York,1988.

Pissanetzky, S. *Sparse Matrix Technology* .Acamedic press INC (London) ltd.1984.

Reddy, J.N. and Gartling, D.K. *The Finite Element Method in Heat Transfer and Fluid Dynamics.*CRC press,1994.

Upadhyay, C. and Manda, S. *Parallelization of Conjugate Gradient Method.*Pr o ceedings of the center for Development of Advanced Computing, India .August 1988 - July 1991.

APPENDIX A
GALERKIN FINITE ELEMENT METHOD

The finite element method is a computational technique for the solution of differential and integral equations that arise in various fields of engineering and applied science.

The major steps in the finite element analysis of a typical problem are as follows:

1. Discretization of the domain into a set of finite elements i.e. mesh generation and node numbering.
2. Weighted-residual (weak) formulation of the governing differential equation.
3. Development of the finite element matrix equations of the problem using the weighted residual form.
4. Assembly of element-level matrix equation to obtain the global system of algebraic equations.
5. Imposition of boundary conditions into the system of algebraic equations.
6. Solution of the system of algebraic equations.
7. Post-processing of the solution and determination of global quantities of interest.

Consider the problem of finding the steady state temperature T(x,y) distribution in a two dimensional medium Ω whose boundary is $\partial\Omega$. The equation governing the temperature distribution is,

$$-\frac{\partial}{\partial x}\left(k\,\frac{\partial T}{\partial x}\right) - \frac{\partial}{\partial y}\left(k\,\frac{\partial T}{\partial y}\right) = Q \text{ in } \Omega \tag{A.1}$$

The boundary conditions of the problem are

$$T = \hat{T} \text{ on } \partial\Omega_T \tag{A.2}$$

$$k \frac{\partial T}{\partial x} \cdot \eta_x + k \frac{\partial T}{\partial y} \eta_y + q_c = q \text{ on } \partial\Omega_q \qquad (A.3)$$

where $\partial\Omega_T$ and $\partial\Omega_q$ are disjoint but continuous portions of the boundary $\partial\Omega$. q_c refers to convective component of heat transfer.

The weak form of the above equation is developed over the typical element Ω^e. There are three steps in the development of a weak form. The first step is to take all non-zero expressions in the equation to one side of equality, multiply the resulting equation (A.1) with a weight function w and integrate the equation over the element domain Ω^e. Hence we get,

$$0 = \int_{\Omega^e} w \left[- \frac{\partial}{\partial x} \left(k \frac{\partial T^e}{\partial x}\right) - \frac{\partial}{\partial y} \left(k \frac{\partial T^e}{\partial y}\right) - Q \right] dx\, dy \qquad (A.4)$$

The expression in the square brackets of the above equation represents a residual of the approximation of the differential equation (A.1) because $T^e(x,y)$ is only an approximation of T(x,y). Therefore equation (A.4) is called the weighted residual statement of Equation (A.1). For every choice of the weight function w(x,y), we obtain an algebraic equation from equation (A.4) among the nodal values T^e_j. For independent choices of w, we obtain a set of **n** linearly independent algebraic equations. This set is called as the weighted-residual finite element model.

The second step consists of distributing the differentiation among **T** and **w** equally so that both **T** and **w** are required to be differentiable only once with respect to x and y. To achieve this Equation (A.4) is integrated by parts. The following identities can be used at this stage.

For differentiable functions w(x,y), $F_1(x,y)$ and $F_2(x,y)$

$$\frac{\partial}{\partial x}(wF_1) = \frac{\partial w}{\partial x}F_1 + w\frac{\partial F_1}{\partial x} \tag{A.5}$$

$$\frac{\partial}{\partial y}(wF_2) = \frac{\partial w}{\partial y}F_2 + w\frac{\partial F_2}{\partial y} \tag{A.6}$$

From divergence theorem

$$\int_{\Omega^e} \frac{\partial}{\partial x}(w\,F_1)\,dx\,dy = \int_{\partial\Omega^e} (wF_1)\,\eta_x\,ds \tag{A.7}$$

$$\int_{\Omega^e} \frac{\partial}{\partial y}(w\,F_2)\,dx\,dy = \int_{\partial\Omega^e} (wF_2)\,\eta_y\,ds \tag{A.8}$$

using equation (A.5), (A.6), (A.7) and (A.8) in (A.4) we obtain

$$0 = \int_{\Omega^e} \left(\frac{\partial w}{\partial x}\cdot k\frac{\partial T}{\partial x} + \frac{\partial w}{\partial y}\cdot k\frac{\partial T}{\partial y} - wQ\right) dx\,dy$$
$$- \int_{\partial\Omega^e} w\left(k\frac{\partial T}{\partial x}\eta_x + k\frac{\partial T}{\partial y}\eta_y\right) ds \tag{A.9}$$

From inspection of the boundary term in Equation (A.9) we note the specification of T constitutes the essential boundary condition. The specification of the coefficient of the weight function in the boundary integral as

$$q_n = k\frac{\partial T}{\partial x}\eta_x + k\frac{\partial T}{\partial y}\eta_y \tag{A.10}$$

Constitutes a natural boundary condition. Here q_n denotes the heat flux normal to the boundary of the element.

The third and last step of the formulation is to use the value of heat flux (A.10) in Equation (A.9) and write as

$$0 = \int_{\Omega^e} \left(\frac{\partial w}{\partial x} \cdot k \frac{\partial T}{\partial x} + \frac{\partial w}{\partial y} \cdot k \frac{\partial T}{\partial y} - wQ \right) dx\, dy - \int_{\partial\Omega^e} w\, q_n\, ds \qquad (A.11)$$

or

$$B\,(w,T) = L(w) \qquad (A.12)$$

where the bilinear form B(.,.) and the linear form L(.) are defined by

$$B(w,T) = \int_{\Omega^e} \left(k \frac{\partial w}{\partial x} \cdot \frac{\partial T}{\partial x} + k \frac{\partial w}{\partial y} \cdot \frac{\partial T}{\partial y}\right) dx\, dy$$

$$L(w) = \int_{\Omega^e} w\, Q\, dx\, dy + \int_{\partial\Omega^e} w\, q_n\, ds$$

When the weight functions w(x,y) is chosen identical to the basis functions of the dependent variable the weighted method is called as the Babnov-Galerkin method.

Grid Generation

The whole domain is divided into smaller triangular element each having six nodes. The elements are chosen as iso-parametric. This allows exact representation of circular boundaries of the contact areas. The grid is also generated within as well as outside the contact area, but with the shape of the obstacles preserved.

Shape Functions

For a six noded standard triangular element with base and altitude equal to unity, the shape functions are

$N_1 = (1 - x - y)(2(1 - x - y) - 1)$

$N_2 = x(2x - 1)$

$N_3 = y(2y - 1)$

$N_4 = 4y(1 - x - y)$

$N_5 = 4x(1 - x - y)$

$N_6 = 4xy.$

If the value of the function is prescribed at the boundary nodes as $f_1 \ldots f_m$ the function value at any other point within the triangular region enclosed by the boundary can be obtained as

$$f(x,y) = \sum_{i=1}^{M} f_i N_i$$

If f is specified as the x or y co-ordinates of the nodes in the physical plane then the interpolation functions is used as mapping function between the physical plane and the normalized region. This mapping is quite important since **FEM** calculations involve integral evaluations. Numerical quadratures for integration are normally specified over the standard triangular region.

Numerical Integration

The integrals which appear in **FEM** calculations are evaluated using Gaussian Quadrature. The gauss points and weight functions used in the present study are listed below.

$\xi = 0,\ 1,\ 0,\ 0,\ 0.5,\ 0.5,\ 0.3333$

$\eta = 0,\ 0,\ 1,\ 0.5,\ 0,\ 0.5,\ 0.3333$

$w = 1/40,\ 1/40,\ 1/40,\ 1/15,\ 1/15, 1/15,\ 1/4.4444.$

APPENDIX B

ELEMENT STIFFNESS MATRIX EQUATION FOR UNSTEADY NONLINEAR HEAT CONDUCTION

The governing equation for transient conduction in two dimensions in dimensionless form is

$$\nabla \cdot k\nabla T = \frac{\partial T}{\partial t} \qquad \text{(B.1)}$$

where we now allow conductivity to depend on temperature i.e.

$$k = k(T) \qquad \text{(B.2)}$$

The linear problem k = constant arises as a special case of Equation B.2.The Galerkin formulation proceeds along following lines. We expand T over an element as $T = \Sigma\, T_j N_j$. Multiplying the governing equation by N_i and integrating over the element we get the equation in matrix form as

$$[A^e]\,[\dot{T}] + [K^e]\,[T] = [q^e] \qquad \text{(B.3)}$$

where

$$[A^e] = \int_{\Omega^e} N_i N_j \, dx\, dy \qquad \text{(B.4)}$$

$$[K^e] = \int_{\Omega^e} k \left[\frac{\partial N_i}{\partial x} \cdot \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \cdot \frac{\partial N_j}{\partial y} \right] dx\, dy \qquad \text{(B.5)}$$

$$[q^e] = \int_{\partial\Omega^e} N_i \cdot k \cdot \frac{\partial T}{\partial n} \cdot ds \qquad \text{(B.6)}$$

$$\dot{T} = \frac{\partial T}{\partial t} \qquad \text{(B.7)}$$

Equation (B.3) can be written as

$$A_{ij} \frac{T_j^{n+1} - T_j^n}{\Delta t} + K_{ij} \cdot T_j^{n+1} = q_i \qquad \text{(B.8)}$$

n and n+1 represents the time interval.

Equation (B.8) can be written in implicit form as

$$\left[A_{ij} + \Delta t \; . \; K_{ij} \right] T_j^{n+1} = A_{ij} . T_j^n + q_i . \Delta t \qquad \text{(B.9)}$$

Thus,

$$T_j^{n+1} = \left[A_{ij} + \Delta t \; . \; K_{ij} \right]^{-1} \left[A_{ij} \; . \; T_j^n + q_i . \; \Delta t \right] \qquad \text{(B.10)}$$

APPENDIX C
ELEMENT STIFFNESS MATRIX EQUATION FOR OIL FLOW IN ROCK FRACTURES

The equation which governs the flow of oil in a rock fracture as derived in Chapter 5 is given by,

$$\frac{\partial}{\partial x}\left[\frac{\partial p_o}{\partial x}\right]^{1/n} + \frac{\partial}{\partial y}\left[\frac{\partial p_o}{\partial y}\right]^{1/n} = \frac{\xi_o}{\left(\frac{n}{2n+1}\right)^{1}\left(\frac{1}{K_o}\right)^{1/n}\left(h^{(n+1)/n}\right)} \cdot \frac{\partial p_o}{\partial t} \qquad (C.1)$$

For the Galerkin formulation the pressure **p** is expanded over the element as $p_o = \Sigma N_j p_{oj}$. The equation (C.1) is multiplied by the shape function N_i and after integrating over the element we get the equation in matrix form as

$$[A^e] \{\dot{p}_o\} + [k^e] \{p_o\} = \{q\} \qquad (C.2)$$

$$A^e_{ij} = \int_{\Omega^e} \beta \cdot N_i N_j \, dx\, dy \qquad (C.3)$$

$$K^e_{ij} = \int_{\Omega^e} \left[\left(\frac{\partial p_o}{\partial x}\right)^{(1-n)/n} \cdot \frac{\partial N_i}{\partial x} \cdot \frac{\partial N_j}{\partial x} + \left(\frac{\partial p_o}{\partial y}\right)^{(1-n)/n} \cdot \left(\frac{\partial N_i}{\partial y} \cdot \frac{\partial N_j}{\partial y}\right)\right] dx\, dy \qquad (C.4)$$

$$q^e_i = \int_{\partial\Omega^e} N_i \left[\left(\frac{\partial p_o}{\partial x}\right)^{1/n} \eta_x + \left(\frac{\partial p_o}{\partial y}\right)^{1/n} \eta_y\right] ds \qquad (C.5)$$

$$\dot{p} = \frac{\partial p}{\partial t} \qquad (C.6)$$

$$\beta = \frac{\xi_o}{\left(\frac{n}{2n+1}\right)\left(\frac{1}{K_o}\right)^{1/n}\left(h^{(n+1)/n}\right)} \qquad (C.7)$$

Equation (C.2) is written as

$$A_{ij}\,\frac{p^{n+1}_{oj} - p_{oj}}{\Delta t} + K_{ij} \cdot p^{n+1}_{oj} = q_i \qquad (C.8)$$

In implicit form the equation (C.8) becomes

$$\left[A_{ij} + \Delta t \, . \, K_{ij}\right] p_{oj}^{n+1} = A_{ij} \, . \, p_{oj} + q_i \, \Delta t \qquad \text{(C.9)}$$

$$p_{oj}^{n+1} = \left[A_{ij} + \Delta t \, . \, K_{ij}\right]^{-1} \left[A_{ij} \, P_{oj}^{n} + q_i \Delta t\right] \qquad \text{(C.10)}$$

Visco Elastic Fluid

The equation which governs the flow of oil with shear stress is given by

$$\frac{\partial}{\partial x}\left[\left(\frac{\partial p_o}{\partial x}\right)^{(1-n)/n} \left[h - \tau_o \,/(\partial p_o/\partial x)\right]^{(2n+1)/n} \right] \frac{\partial p_o}{\partial x}$$

$$+ \frac{\partial}{\partial y}\left[\left(\frac{\partial p_o}{\partial x}\right)^{(1-n)/n} \left[h - \tau_o \,/(\partial p_o/\partial y)\right]^{(2n+1)/n} \right] \frac{\partial p_o}{\partial y} = \beta \, . \, \frac{\partial p_o}{\partial t} \qquad \text{(C.11)}$$

where

$$\beta = \frac{\xi_o . h}{\left(\frac{n}{2n+1}\right)\left(\frac{1}{K_o}\right)^{1/n}} \qquad \text{(C.12)}$$

The Galerkin FEM formulation gives the following matrix equation:

$$[B^e] \, \{\dot{p}_o\} + [G^e] \, \{p_o\} = \{r^e\} \qquad \text{(C.13)}$$

where

$$B_{ij}^e = \int_{\Omega^e} \beta . \, N_i N_j \, dx \, dy \qquad \text{(C.14)}$$

$$G_{ij}^e = \int_{\Omega^e} \left[\left(\frac{\partial p_o}{\partial x}\right)^{(1-n)/n} \left[h - \tau_o/(\partial p_o/\partial x)\right]^{(2n+1)/n} . \frac{\partial N_i}{\partial x} . \frac{\partial N_j}{\partial x} \right.$$

$$\left. + \left(\frac{\partial p_o}{\partial y}\right)^{(1-n)/n} \left[h - \tau_o \,/(\partial p_o/\partial y)\right]^{(2n+1)/n} . \frac{\partial N_i}{\partial y} . \frac{\partial N_j}{\partial y} \right] dx \, dy$$

(C.15)

$$r_i^e = \int \left[\left(\frac{\partial p_o}{\partial x}\right)^{1/n} \left[h - \tau_o/(\partial p_o/\partial x)\right]^{(2n+1)/n} \eta_x \right.$$

$$\partial\Omega^e$$

$$+ \left(\frac{\partial p_o}{\partial y}\right)^{1/n} \left[h-\tau_o/(\partial p_o/\ \partial y)\right]^{(2n+1)/n} \eta_y \Bigg] ds \qquad (C.16)$$

$$\dot{p}_o = \frac{\partial p_o}{\partial t} \qquad (C.17)$$

The implicit schemes gives the following matrix equation:>

$$[B_{ij} + \Delta t \ . \ G_{ij}]\ p_{oj}^{n+1} = B_{ij} .\ P_{oj}^{n} + r_i \ . \ \Delta t \qquad (C.18)$$

$$p_{oj}^{n+1} = \left[B_{ij} + \Delta t \ . \ G_{ij} \right]^{-1} \left[B_{ij} \ . \ P_{oj}^{n} + r_i \ \Delta t \right] \qquad (C.19)$$

APPENDIX D

DERIVATION OF THE DIFFERENTIAL EQUATIONS GOVERNING OIL-WATER FLOW IN A SINGLE ROCK FRACTURE

The equations governing oil and water pressures in two component flow through a fracture medium is derived here.

Assumptions

(1) Two fluid phases, namely oil and water flow simultaneously.

(2) The fluids are immicible i.e. there is no mass transfer between them.

(3) Gravity term is neglected in the momentum equation. It is reasonable because the time frames are short and as a result the gravity override effect will not show up.

(4) Relative permeabilities of phases and capillary pressure are only a function of saturation.

(5) There is no net mass source in the flow field.

Derivation

Following are the basic equations which are used to derive the final pressure equations for flow. By solving Stokes equation for flow between two walls we have the oil and water flow velocities as

Oil

$$u_o\,(x,y) = -\,\alpha.K_{ro}\,\left(\frac{\partial p_o}{\partial x}\right)^{1/n} \qquad \text{(D.1)}$$

$$v_o\,(x,y) = -\,\alpha.K_{ro}\,\left(\frac{\partial p_o}{\partial y}\right)^{1/n} \qquad \text{(D.2)}$$

$$\alpha = \left(\frac{n}{2n+1}\right) \left(h^{(n+1)/n}\right) \left(\frac{1}{K_o}\right)^{1/n} \qquad (D.3)$$

Water

$$u_w(x,y) = -\beta \,.\, K_{rw} \left(\frac{\partial p_w}{\partial x}\right)^{1/n} \qquad (D.4)$$

$$v_w(x,y) = -\beta \,.\, K_{rw} \left(\frac{\partial p_w}{\partial y}\right)^{1/n} \qquad (D.5)$$

$$\beta = \left(\frac{n}{2n+1}\right) \left(h^{(n+1)/n}\right) \left(\frac{1}{K_w}\right)^{1/n} \qquad (D.6)$$

Mass balance

The mass balance equation for each of the components is given below:

$$\text{Oil :} \quad \frac{\partial(S_o\rho_o)}{\partial t} + \nabla \,.\, u_o \,.\, \rho_o = q_o \qquad (D.7)$$

$$\text{Water:} \quad \frac{\partial(S_w\rho_w)}{\partial t} + \nabla \,.\, u_w \,.\, \rho_w = q_w \qquad (D.8)$$

$$S_o + S_w = 1 \qquad (D.9)$$

In the present work q_o and q_w, the mass sources are taken to be absent.

Equation of State

The following quantities are taken to be local functions and independently specified.

Compressibility   Volumetric thermal expansion coefficient

Oil

$$\xi_o = \frac{1}{\rho_o} \cdot \frac{\partial \rho_o}{\partial p_o}\bigg|_T \qquad \beta_o = -\frac{1}{\rho_o} \cdot \frac{\partial \rho_o}{\partial T}\bigg|_{P_o} \qquad (D.10)$$

Water

$$\xi_w = \frac{1}{\rho_w} \cdot \frac{\partial \rho_w}{\partial p_w}\bigg|_T \qquad \beta_w = -\frac{1}{\rho_w} \cdot \frac{\partial \rho_w}{\partial T}\bigg|_{P_w} \qquad (D.11)$$

Constitutive relations

The following constitutive relations are required to close the system of equation. These must be generated from field experiments.

$$p_c = p_o - p_w = p_c(S_w) = \frac{1 - S_w}{S_w} \quad \text{(D.12)}$$

$$K_{rw} = K_{rw}(S_w) = 0.3\, S_w \quad \text{(D.13)}$$

$$K_{ro} = 1 - \exp\left(-\frac{1}{S_w}\right) \quad \text{(D.14)}$$

Substituting equation of state and constitutive relations into mass balance equation one gets pressure equation for oil and water

$$\frac{\partial p_o}{\partial t}\left[S_o \cdot \xi_o - \frac{dS_w}{dp_c}\right] + \frac{dS_w}{dp_c} \cdot \frac{\partial p_w}{\partial t} - S_o\beta_o \cdot \frac{\partial T}{\partial t} = -\nabla . u_o \quad \text{(D.15)}$$

$$\frac{\partial p_w}{\partial t}\left[S_w \cdot \xi_w - \frac{dS_w}{dp_c}\right] + \frac{dS_w}{dp_c} \cdot \frac{\partial p_o}{\partial t} - S_w\beta_w \cdot \frac{\partial T}{\partial t} = -\nabla . u_w \quad \text{(D.16)}$$

In the present study, flow is assumed to be isothermal substituting the velocity equations (D.1), (D.2), (D.4), (D.5) into (D.15) and (D.16) one gets the final form of pressure equation for oil and water as given below

$$\frac{\partial p_o}{\partial t}\left[S_o\xi_o - \frac{dS_w}{dp_c}\right] + \frac{dS_w}{dp_c} \cdot \frac{\partial p_w}{\partial t} = \propto \left[\frac{\partial}{\partial x}\left(K_{ro}\left(\frac{\partial p_o}{\partial x}\right)^{1/n}\right) + \frac{\partial}{\partial y}\left(K_{ro}\left(\frac{\partial p_o}{\partial y}\right)^{1/n}\right)\right] \quad \text{(D.17)}$$

$$\frac{\partial p_w}{\partial t}\left[S_w\xi_w - \frac{dS_w}{dp_c}\right] + \frac{dS_w}{dp_c} \cdot \frac{\partial p_o}{\partial t} = \propto \left[\frac{\partial}{\partial x}\left(K_{rw}\left(\frac{\partial p_w}{\partial x}\right)^{1/n}\right) + \frac{\partial}{\partial y}\left(K_{rw}\left(\frac{\partial p_w}{\partial y}\right)^{1/n}\right)\right] \quad \text{(D.18)}$$

The Galerkin formulation for oil pressure equation proceeds along the following lines. p_o is expanded over an element as $p_o = \Sigma p_{oj} N_j$. Multiplying the governing equation by N_i and integrating over the element we get

$$\sum_{j=1}^{n_e} \left[A^e_{ij} \frac{dp^e_{oj}}{dt} + K^e_{ij}\, p^e_{oj} \right] + Q^e_i - q^e_i = 0 \qquad \text{(D.19)}$$

In the matrix form it can be written as

$$[A^e] \; \{\dot{p}_o\} + [K^e]\{p_o\} = \{q^e\} - \{Q^e\} \qquad \text{(D.20)}$$

where a superposed dot on p_o denotes a derivative with time and

$$A^e_{ij} = \int_{\Omega^e} (S_o \, \xi_o - \frac{dS_w}{dp_c}) \,.\, N_i . N_j \,.\, dx\, dy \qquad \text{(D.21)}$$

$$K^e_{ij} = \int_{\Omega^e} \propto \left[\left\{ K_{ro} \left(\frac{\partial p_o}{\partial x}\right)^{(1-n)/n} . \frac{\partial N_i}{\partial x} . \frac{\partial N_j}{\partial x} \right\} + \left\{ K_{ro} \left(\frac{\partial p_o}{\partial y}\right)^{(1-n)/n} . \frac{\partial N_i}{\partial y} . \frac{\partial N_j}{\partial y} \right\} \right] dx\, dy \qquad \text{(D.22)}$$

$$Q^e_i = \int_{\Omega^e} \left[\frac{dS_w}{dp_c} \cdot \frac{\partial p_w}{\partial t} \right] . \, N_i \; dx\, dy \qquad \text{(D.23)}$$

$$q^e_i = \int_{\Omega^e} \propto \left[K_{ro} \left(\frac{\partial p_o}{\partial x}\right)^{1/n} \eta_x + K_{ro} \left(\frac{\partial p_o}{\partial y}\right)^{1/n} \eta_y \right] ds \qquad \text{(D.24)}$$

Similarly the Galerkin formulation is carried out for the water pressure equation. The matrix form of the pressure equation is written as

$$[B^e] \; \{\dot{p}_w\} + [L^e] \; \{p_w\} = \{r^e\} - \{R^e\} \,. \qquad \text{(D.25)}$$

where

$$B^e_{ij} = \int_{\Omega^e} \left[S_w \cdot \xi_w - \frac{dS_w}{dp_c} \right] N_i N_j \; dx\, dy \qquad (D.26)$$

$$L^e_{ij} = \int_{\Omega^e} \beta \cdot \left[(K_{rw} \left(\frac{\partial p_w}{\partial x}\right)^{(1-n)/n} \cdot \frac{\partial N_i}{\partial x} \cdot \frac{\partial N_j}{\partial x}) + (K_{rw} \left(\frac{\partial p_w}{\partial y}\right)^{(1-n)/n} \cdot \frac{\partial N_i}{\partial y} \cdot \frac{\partial N_j}{\partial y} \right] \qquad (D.27)$$

$$R^e_i = \int_{\Omega^e} \left(\frac{dS_w}{dp_c} \cdot \frac{\partial p_o}{\partial t}\right) N_i \cdot dx\, dy \qquad (D.28)$$

$$r^e_i = \int_{\Omega^e} \beta \left(K_{rw} \left(\frac{\partial p_w}{\partial x}\right)^{1/n} \eta_x + K_{rw} \left(\frac{\partial p_w}{\partial y}\right)^{1/n} \eta_y \right) ds \qquad (D.29)$$

$$\dot{p}_w = \frac{\partial p_w}{\partial t} \qquad (D.30)$$

The two matrix equation (D.20) and (D.25) are solved by implicit finite difference scheme. The pressure p_o and p_w at time n+1 is given by

$$p_o^{n+1} = \left[A_{ij} + \Delta t\; K_{ij}\right]^{-1} \left[\Delta t \cdot q_i - \Delta t \cdot Q_i + A_{ij}\, p_o^n \right] \qquad (D.31)$$

$$p_w^{n+1} = \left[B_{ij} + \Delta t\; L_{ij}\right]^{-1} \left[\Delta t \cdot r_i - \Delta t \cdot R_i + B_{ij}\, p_w^n \right] \qquad (D.32)$$

Equation (D.31) and (D.32) are coupled equations due to terms

$$\frac{dS_w}{dp_c}, \; \frac{dS_w}{dp_c} \cdot \frac{\partial p_w}{dp_c}, \; \frac{dS_w}{dp_c} \cdot \frac{\partial p_o}{\partial t} .$$

Therefore they are simultaneously solved. First Equation (D.31) is solved for p_o^{n+1}; this value is plugged in Equation (D.32) to get p_w^{n+1}. Then p_w^{n+1} is again plugged in (D.31). This continues till convergence of p_o and p_w is achieved within a time step. The equations are subsequently solved for next time step. The approach of assigning two unknowns per node (namely p_o and p_w) will be taken up in the future.

ME-1895-M-SHE-DEV